VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
EKONOMICKÁ FAKULTA

KATEDRA APLIKOVANÉ INFORMATIKY

Racionalizace internetových informačních služeb

Rationalization of Internet Information Services

Student: Daniel Dimitrov
Vedoucí bakalářské práce: Ing. Ministr Jan, Ph.D.

Ostrava 2010

VŠB - Technická univerzita Ostrava
Ekonomická fakulta
Katedra aplikované informatiky

Zadání bakalářské práce

Student:	Daniel Dimitrov
Studijní program:	B6209 Systémové inženýrství a informatika
Studijní obor:	6209R001 Aplikovaná informatika
Téma:	Racionalizace internetových služeb
	Racionalization of Internet Services

Zásady pro vypracování:

1. Úvod
2. Metodologická východiska a vymezení pojmů v oblasti internetových služeb
3. Analýza stávajícího stavu internetových služeb vzhledem k uživatelským potřebám
4. Návrh řešení
5. Závěr

Seznam použité literatury

Seznam zkratek

Prohlášení o využití výsledků bakalářské práce

Přílohy

Seznam doporučené odborné literatury:

CASTRO, E. HTML, XHTML a CSS: Názorný průvodce tvorbou WWW stránek. Brno: Computer Press, 2007. 440 s. ISBN 978-80-251-1531-2.

KOSEK, J. PHP: Tvorba interaktivních internetových aplikací. Praha: Grada Publishing a.s., 1999. 492 s. ISBN 80-7169-373-1.

SHELDON, R. SQL - začínáme programovat. Praha: Grada Publishing, spol. s r.o., 2005. 500 s. ISBN 80- 247-0999-6.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce:	Ing. Jan Ministr, Ph.D.
Datum zadání:	20.11.2009
Datum odevzdání:	07.05.2010

Místopřísežně prohlašuji, že jsem celou práci, včetně všech příloh, vypracoval samostatně

Datum : 30.4.2010 *Podpis:*

Rád by jsem vyjádřil mimořádné poděkování mému vedoucímu bakalářské práce Ing. Janu Ministrovi, Ph.D., zejména za trpělivost, ochotu, odborné vedení a cenné rady, které mi poskytl.

Obsah

1 Úvod

Při výběru tématu své bakalářské práce jsem dlouho váhal nad výběrem tématu. Věděl jsem, že téma by se mělo týkat programování a správy databází. Protože kvůli těmto oborům jsem studoval střední a následně i vysokou školu a tudíž jsem měl pro tyto témata předpoklady a nabyté znalosti. Rovněž programovacích jazyků a databázových systémů existuje celá řada. Některé jsem se učil, jiné ne. Uvažoval jsem o tématu bakalářské práce kombinující programovací jazyk a databázi, které jsem se nikdy neučil. Tato kombinace mně napadla proto, že bych prohloubil své programátorské znalosti v jiném jazyku a praktické zkušenosti v jiném databázovém systému. Navíc bych získal větší přehled o dostupných jazycích a možnost porovnávat mezi nimi. Prostě jsem toužil po nepoznané kombinaci programovacího jazyka a databáze. Takže ještě před zadáním bakalářské práce jsem měl cíl vypracovat práci, která by mě bavila, něco nového naučila a ve které bych zužitkoval své dlouhodobé znalosti v oborech programování a správa databází.

Na poslední chvíli jsem dostal nabídku tématu velice zajímavou a přesně padnoucí do „mého“ zadání. Šlo o zadání výstavby informačního systému pro tvorbu a sdílení citací pro registrovanou skupinu uživatelů moderovanou zadavatelem bakalářské práce, panem Ing. Janem Ministrem.

Padnoucím do „mého“ zadáním v tom smyslu, že zadavatel požadoval při tvorbě projektu použít programovací jazyk PHP, určený pro tvorbu dynamických stránek, a multiplatformní databázový systém MySQL. Tedy technologie mnou ještě nepoznané, na světě velmi rozšířené a pro mě ve výsledku velmi zajímavé.

1.1 Zadání projektu

Samotné zadání projektu bylo:

Vytvořit informační systém pro správu citací registrovaných autorů s možností prezentace těchto citací. Systém musí splňovat aspekty spolehlivosti, rychlosti a bezpečnosti. Zadání obsahovalo i administraci systému a další eventuální vývoj.

Vlastnosti budoucího IS byly definovány zadavatelem takto:

- Registrace schvalovaná moderátorem,
- Profil jednotlivých autorů,
- Možnost autora vkládání, upravování a mazání svých citací,
- Nástěnka příspěvků sdílena jen registrovanými uživateli systému,
- Vyhledávání příspěvků dle kriterií,
- Nástěnka konferencí vedená administrátorem,
- Možnost smazání nebo dočasnému vypnutí uživatele,
- Veškerá data i operace vhodně zabezpečená,
- Řádně zabezpečit vstupní data do databází,
- Přehledné a rychlé GUI systému.

Použité technologie:

- programovací jazyk PHP na tvorbu dynamických webových stránek,
- použití databázového systému MySQL,
- spolupráce s webovým serverem Apache,
- HTML, CSS, JavaScript.

Práce bude realizována jako systémová integrace výše zmíněných technologií v jeden funkční celek. Tedy úkolem bude co nejlépe zakomponovat použité prvky systému tak, aby vznikl informační systém dle zadání zadavatele.

1.2 Cíl projektu

Samotný cíl projektu je vytvořit bezpečný, kvalitní, rychlý, přehledný, spolehlivý a užitečný informační systém, obsahující vše potřebné pro svůj bezchybný provoz. Prioritou systému je bezpečnost dat a spolehlivost systému. Dále se předpokládá podpora systému administrátorem pro další rozvoj, pomoc uživatelům a dohled nad systémem.

2 Metodologická východiska, vymezení pojmů

2.1 Systémová integrace

Systémová integrace představuje funkční spojení různých komponent, procesů ve firmě a samostatných systémů. Pomocí systémové integrace dochází k vytvoření funkčního provázaného celku vytvořeného z různých komponent, technologií a výrobců. Přičemž se upřednostňuje použití komponent stejného výrobce a pokud možno podobné či stejné technologie. Při integraci systému hraje důležitou roli i ekonomika výstavby systému. [6]

Systémové integrace je hlavně tvořena [6]:

- integrací vizí o IS/IT ve firmě,
- integrací interních firemních procesů,
- integrací datovou (používání společných dat je důležité),
- integrací organizace s okolím (důležité při externí komunikací firmy).

Komponenty jsou nedílnou součástí systémové integrace, která tyto komponenty spojuje, nastavuje a integruje do jednoho funkčního systému. Komponenty jsou systémy menších rozměrů, jejichž priorita vzroste díky komunikaci s ostatními integrovanými komponenty. Komponenty komunikují mezi sebou nebo s řídící jednotkou pomocí interface (rozhraní). Komunikace komponent je opět řešena systémovou integrací. Důležitým kriteriem pro výstavbu systému je použití jednotných dat uvnitř vybudovaného systému, aby komunikace jednotlivých komponent fungovala efektivně a rychle. Součásti integrace je i řešení nežádoucích stavů, tedy musí být definován přesně stanovený postup instrukcí při vzniku chyby či anomálie. Systém musí počítat s určitou chybovostí systému, prostředí nebo lidského faktoru. Je vždy lepší, když systém informuje o výskytu chyby chybovou hláškou, nežli akceptovat výpadek systému. [6]

Jako úspěch integrace se bere minimálně jedna z těchto hodnot [6]:

- snížení nákladů firmy,
- lepší přehlednost, průhlednost a kontrolovatelnost systému jako celku pro řízení firmy,
- zvýšení síly a konkurenceschopnosti firmy,
- zvýšení bezpečnosti systému díky integraci,
- možnost propojení interního systému s několika dalšími interními systémy,
- možnost systému v budoucnu lehce aktualizovat či rozšiřovat.

Při realizaci mohou vzniknout i některé z těchto chyb [6]:

- zahájení projektu bez jasné strategie,
- požadavky na informační systém nejsou v souladu s potřebami organizace,
- nepřipravenost zaměstnanců na odlišný styl práce,
- touha po nejlevnějším řešení.

Vznik i některých z těchto chyb vede k špatnému zavedení systému. Takže ve výsledku nejen že nebude inovace IS vyhovovat potřebám firmy, ale může integrace způsobit vysoké náklady při nulové efektivitě. [6]

Systémová integrace se opírá o kvalitní spolupráci informačního manažera a systémového integrátora. [6]

2.2 Informační manažer

Informační manažer je členem vrcholového managementu firmy a má zodpovědnost za vývoj informačního systému a informačních a komunikačních technologií. Vykonává důležité manažerské funkce při správě IS/IT, spravuje rozpočet pro vývoj IS/IT a připravuje informační strategii organizace. [6]

Informační manažer také zajišťuje výběr vhodného dodavatele IS/IT a při budování informačních systémů spolupracuje se systémovým integrátorem. Kvalitní spolupráce zaručuje efektivní systémovou integraci. [6]

2.3 Systémový integrátor

Systémový integrátor představuje dodavatele služby systémové integrace. Jedná se o jednotlivce či firmu zajišťují dodávku služeb integrace.

Mezi obecné předpoklady integrátora patří [6]:

- široká škála znalostí z různých oborů,
- vypracování plánu systémové integrace,
- odpovědnost za kompletní a kvalitní integraci systému,
- orientace ve více technologií.

Obecně lze říct jak je kvalitní systémový integrátor tak je kvalitní celý integrační proces. [6]

2.4 LAMP

Další teoretická část věnovaná softwarovým komponentům, které budou při vývoji systému použity. Touto kapitolou začne popis pojmů, ale i technologií, které budou využity při práci na projektu.

Zkratka LAMP označuje sadu svobodného softwaru, která se používá pro implementaci dynamických webových stránek. LAMP značí základní skladbu softwaru pro použití na serverech (operační systém se základní skladbou programů). Při rozložení zkratky LAMP zjistíme, co všechno tato hierarchie nabízí. [19]

Zkratka LAMP obsahuje soubor těchto softwarových komponent [19]:

- Linux (GNU/Linux) – operační systém,
- Apache – webový server,
- MySQL – databázový systém,
- PHP – programovací jazyk.

Takže lze soubor LAMP označit jako systém vytvořený softwarovou integrací komponent s bezvadnou komunikací mezi nimi. Jednotlivé softwarové komponenty se vyvíjejí samostatně různými výrobci s ohledem na ostatní prvky systému LAMP. Nicméně jednotlivé komponenty mohou být zaměněny za jiné a tak hierarchii LAMP lze jednoduše měnit. Například databázový systém MySQL lze nahradit open-source databází

PostgreSQL (LAPP). Nebo místo jazyka PHP lze použít Python či Perl. Při této obměně se mění i zkratka hierarchie dle počátečních písmen jednotlivých technologií. Avšak hierarchie LAMP (Linux, Apache, MySQL, PHP) je hierarchií nejvíce používanou a na světě nejvíce oblíbenou. [19]

Všechny prvky systému lze označit za open-source software, software s otevřeným zdrojovým kódem. Tento software má velkou výhodu v bezpečnosti, neboť chyby ve zdrojovém kódu může hledat daleko větší skupina lidí, avšak i hackeři mohou na tento kód lehce nahlížet. Ale i přes toto menší riziko je tento druh softwaru označován jako velmi bezpečný, protože „více očí více vidí“. Takže chyby jsou opravovány daleko rychleji, což jeden z podstatných důvodů tak vysokého nasazení hierarchie LAMP na světových serverech. [19]

Mezi další výhody patří [19]:

- malé hardwarové požadavky na server,
- aktuální systém s ohledem na bezpečnost,
- vysoký výkon,
- open source licence.

Při vývoji systému byla použita hierarchie LAMP pro simulaci reálného nasazení systému na webhostingu školy. Nyní budou postupně vysvětleny prvky LAMP hierarchie.

2.5 Linux

Jako první technologie bude popsán samotný běh serveru, tedy operační systém. Vysvětlované pojmy jsou: jádro, GNU, licence GPL a distribuce. [5]

Operační systém GNU/Linux je tvořen samotným jádrem Linux, tzv. kernelem, a balíkem svobodného softwaru GNU. V běžné řeči je často tomuto spojení používáno označení Linux, avšak toto označení patří správně jádru systému. Skladba softwaru systému závisí na každé distribuci a celý takto vytvořený systém je licencován GPL licencí. Některé z těchto pojmů mají i třicetiletou historii. [5]

2.5.1 Historie

Do počáteční historie Linuxu se zapsaly dvě jména Richard Stallman a Linus

Torvald. Richard Stallman byl zakladatelem projektu GNU (vynikl v roce 1983), jehož cílem bylo vytvořit unixový operační systém složený výhradně ze svobodného softwaru. V roce 1990 začal projekt GNU vyvíjet své vlastní jádro od kterého bylo časem upuštěno z důvodů složitého a pomalého vývoje. [5,16]

V roce 1991 bylo představeno jiné a dodnes používané jádro, jehož zakladatelem byl finský student Linus Torvalds. Tento kernel měl označení Linux a prvně představená verze byla 0.01. Tímto došlo ke spojení kernelu od Torvaldse a GNU projektu. A vznikl operační systém GNU/Linux. [16]

Od této doby se na vývoji kernelu podílelo tisíce vývojářů z celého světa včetně zakladatele samotného jádra. Jádro Linuxu se nepoužívá jen v operačních systémech počítačů, ale nachází se ve všech možných druhzích elektroniky – routery, mobilní telefony, kamerové systémy, atd. V těchto menších zařízení se ve většině případů používá starší vývojová větev jádra. [5,16]

2.5.2 Licence

Jádro a GNU komponenty jsou licencovány pod GNU General Public License (GPL). Pokud je software šířen pod touto licencí, znamená to, že je tento software volně šiřitelný. Software může být kýmkoliv provozován, upravován a nadále prodáván s kompletním zdrojovým kódem. Licence GPL také obsahuje stanovení podmínky odpovědnosti. To znamená, že vývojáři nejsou odpovědní za případné škody vzniklé používáním softwaru. [5]

Samozřejmě i tato GPL licence prošla vývojem a tak od založení postupně vyšly 3 verze licence [5]:

- GPLv1 vydaná v lednu 1989,
- GPLv2 vydaná v červnu 1991,
- GPLv3 vydaná v červnu 2007.

Podobně jako u jádra Linuxu, tak také na vývoji pořád osobně pracuje i samotný zakladatel licence, tedy Richard Stallman. [16]

2.5.3 Jádro

Součástí operačního systému je samotné jádro Linux neboli kernel. Toto jádro je základním kamenem každého operačního systému. Jádro se zavádí při startu OS do operační paměti a zůstává v činnosti po celou dobu běhu PC. Jeho selhání zapříčiní selhání běhu celého počítače. [5]

Mezi základní činnosti jádra patří [5]:

- ovládání počítače,
- správa prostředků,
- ovládání hardware.

Samozřejmě činností jádra je daleko více, ale počet těchto činností se odvíjí od druhu jádra.

Jádra mohou být trojího druhu [5]:

- monolitické jádro - poskytující veškeré služby aplikacím,
- mikrojádro - poskytující základní sadu (nejnutnějších) služeb,
- hybridní jádro – kombinuje vlastnosti předchozích jader.

V systému GNU/Linux je použité monolitické jádro, zatímco v sytému Windows (XP, Vista, Seven) je jádro hybridní. [5]

2.5.4 Distribuce

Představuje spojení linuxového jádra a aplikací GNU. Distribuce se liší programovým vybavením, podporou a zaměřením. Naopak distribuce spojuje samotné jádro Linux, které se může lišit verzí kernelu. Distribucí je mnoho, mluví se o cca 450 distribucí, a ty se mohou dělit z různých hledisek. [5]

Možné rozdělení distribucí [5]:

- placené (podnikové verze) nebo zdarma,
- s technickou podporou nebo s podporou linuxové komunity,
- pro domácí PC nebo pro server,
- zaměření distribuce (například distribuce pro netbooky).

Distribuce jsou většinou tvořeny jako úprava již existují distribuce, například dost distribucí vycházejí z Debianu, Fedory nebo Gentoo Linuxu. Distribuce jsou sestavovány jednotlivci, týmy dobrovolníků nebo komerčními firmami. Mezi nejpoužívanější distribuce patří: Debian, SUSE, Ubuntu, Fedora a Madriva. [5]

2.5.5 Serverový operační systém

GNU/Linux představuje velmi kvalitní operační systém pro servery. GNU/Linux pro své výhody je hojně nasazován jako operační systém pro serverové počítače. [5]

Výhody nasazení GNU/Linux na server [5]:

- optimalizované jádro pro verzi server,
- vysoká stabilita a bezpečnost,
- nízké hardwarové nároky, protože verze server většinou nemá GUI a je ovládán přes terminál (příkazovou řádku),
- vždy aktuální OS pomocí pravidelných aktualizací,
- nízké počáteční náklady na zřízení OS,
- Open source software,
- široká podpora file-systémů,
- široká paleta podporovaného hardwaru.

Důležité při výběru distribuce je znát co od distribuce očekáváme, jestli preferujeme aktuálnost nebo stabilitu balíčků, jakou preferujeme skladbu programů či jak kterou distribuci už známe z desktopu. [5]

Rozhodující roli hraje i podpora distribuce, která zabezpečuje aktuální systém. Nové distribuce jsou nabízeny většinou v pravidelných cyklech, u některých to bývá pravidelně půlročně, a přinášejí nový software s novými funkcemi. Ale bezpečnostní podpora každé verze distribuce bývá delší než samotný půlrok. Například Ubuntu nabízí každé dva roky distribuci s označením LTS („Long Term Support“), které představují 3-letou podporu pro desktop a až 5-letou podporu pro server edition. Během této doby jsou vydávány bezpečnostní aktualizace, které nepřidávají nové funkce, ale opravují použitý software na serveru. Tím zvyšují stabilitu systému, vzájemnou kompatibilitu softwaru a spolehlivost serveru. Taková podpora ze strany distributora softwaru ulehčují správcům serverů práci

spjatou s upgradem systému a administrátor má jistotu, že verze jednotlivých programů a služeb na serveru jsou dobře sladěny. [5]

Na druhou stranu jsou pořád vydávány co půl roku nové distribuce, které nabízejí nový software, nabízející větší komfort práce a přidávají nové užitečné funkce. A tím se dostávám k základní otázce této sekce a to jaké je očekávání správce. Před čím dává přednost: aktuálnost nebo stabilita systému? [5]

Pro serverový operační systém je nejdůležitější ho udržovat ve funkčním, bezpečném a aktuálním stavu. Tím slovem bezpečný se míní například správa SW firewallu (mimochodem integrovaném v samotném jádru Linux a nazývaném iptables), přístupových práv, pravidelných záloh systému a preventivní kontrolou hardwaru (například kontrola S.M.A.R.T. hodnot z jednotlivých pevných disků a tím zjištění fyzického zdraví disků). [5]

2.6 Webový server Apache

Samotný serverový operační systém již máme vysvětlený a tvoří základ pro náš LAMP server. Druhým písmenkem ve zkratce LAMP je A jako Apache. Konkrétněji Apache HTTP Server, tedy softwarový webový server. Samozřejmě je tento server Open-Source a má licenci Apache License v2.0 kompatibilní s licencí GPLv3.

Apache HTTP server patří mezi projekty tzv. „Apache Software Foundation-ASF“, což je nezisková společnost registrovaná v Delaware, USA. Tato komunita vývojářů organizuje několik projektů (řádově desítky projektů) psaných v několika programovacích jazycích – C, C++, C#, PHP, Java, XML, JavaScript, Perl, Python, Ruby, SQL. [9]

Mezi tyto projekty mimo jiné patří [9]:

- Apache HTTP Server – webový server,
- Apache Tomcat – Servlet/JSP Container,
- Apache PDFBox – Java knihovna pro práci s PDF dokumenty,
- Apache Pivot – platforma pro tvorbu internetových aplikací psaných v Java/XML,
- mnoho dalších projektů.

Jak jde vidět skupina projektů je hojná a hodně multiplatformní. Zatím nejvíce rozšířeným, oblíbeným a asi i nejdůležitějším projektem ASF skupiny zůstává Apache HTTP Server, jenž je součástí LAMP hierarchie. [9]

Základní funkcí webových serverů je vracení odpovědí na HTTP požadavky od klientů – webových prohlížečů. Odpověď je klientovi ve většině případů zobrazena jako HTML stránka. Odpovědí může být i obrázek či binární soubor. [3,4]

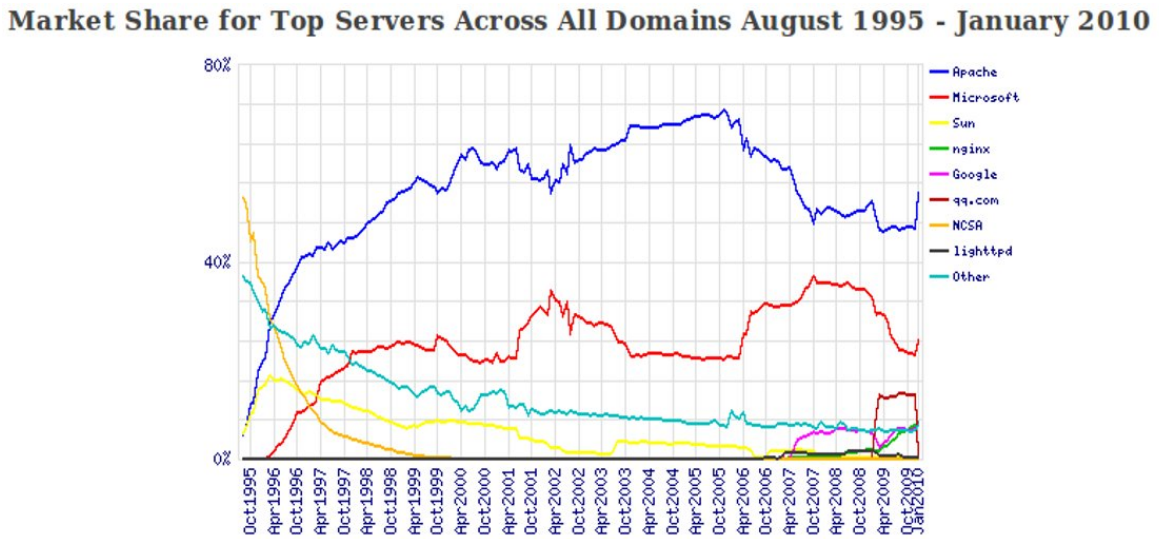
Součástí odpovědi je stavový kód odpovědi. Při korektní odpovědi klientovi je poslán stavový kód 200, značící stav OK. Samozřejmě dochází i k chybám, a to na obou stranách komunikace- klient i server. Tyto chyby jsou serverem detekovány a poslány klientovi ve tvaru HTTP odpovědi, konkrétně jsou chyby vyjádřeny stavovým kódem [3]:

- 3xx- chybné přesměrování,
- 4xx- chybné vyřízení požadavku (špatná syntaxe dotazu),
- 5xx- chyba na straně serveru (přetížení, výpadek).

Webový server má dvě možnosti zpracování odpovědi klientovi [3]:

- posláním statického obsahu určitého souboru (obsah není měněn serverem a je jen přeposlán klientovi),
- zpracování dynamického obsahu (databáze, zpracování HTML stránky určitým programovacím jazykem) a poslání odpovědi ve formě HTML stránky klientovi.

Apache HTTP Server je celosvětově nejoblíbenějším serverem a jeho podíl neustále roste, což také dokazuje server netcraft.com zabývající se statistikami použitých serverů od roku 1995. Následující obrázek je více než výmluvný. [4]



Obrázek 2.1: Statistika rozdělení jednotlivých webových serverů [13]

2.7 MySQL

Software, který pracuje s databázemi se jmenuje databázový server a spolu s webovým serverem tvoří neodmyslitelnou součást LAMP serveru. Databázový server pracuje s nejcennějšími prvky moderního IS – s daty uživatelů. Jejich organizace, uložení a výběr jsou základními funkcemi každého databázového serveru a kvalitní server musí tyto operace provádět bezpečně a spolehlivě. [25]

Databázový systém bývá odborně označován jako SŘBD – Systém řízení báze dat (anglicky DBMS – Database management systém). SŘBD je chápán jako software pro práci s daty (ukládání, správa, mazání dat) a tvoří rozhraní mezi aplikačními programy a samotnými daty. [25]

V našem případě bude použita celosvětově nejoblíbenější Open-Source relační databáze – MySQL.

MySQL je relační databázový systém vytvořený švédskou firmou MySQL AB. Avšak v roce 2008 došlo k akvizici firmy MySQL AB firmou Sun Microsystems, která spravovala databázi do nedávna (leden 2010), kdy došlo odkoupení firmy Sun Microsystems konkurenční firmou ORACLE. [25]

Úspěšná databáze MySQL má vysoký podíl v současně používaných databázích. Tento úspěch je z několika důvodů, kterými jsou multiplatforma, výkonnost a zároveň se jedná o volně šiřitelný software. MySQL je jednak šířena pod bezplatnou licencí GPL, ale také pod placenou komerční licencí, kde stále platí původní licence GPL. [25]

Základní vlastnosti databáze MySQL [25]:

- nezávislost na platformě (Windows, Linux, FreeBSD, Sun Solaris),
- rychlost MySQL,
- uložené procedury,
- podpora triggerů,
- podpora pohledů,
- podpora transakcí.

MySQL podporuje velký výběr datových typů (celočíselné, desetinné, řetězcové a datové). Kompletní přehled podporovaných datových typů databázového systému najdete

na webových stránkách výrobce databáze. [8]

Standardně se MySQL databáze spravuje přes příkazovou řádku, konkrétně příkaz „mysql“ (pro odborníky to představuje nejlepší správu databáze). Existují i softwarové GUI nástroje pro správu MySQL databáze, které nabízejí lepší komfort při práci s daty. [14]

Nástroje pro práci s MySQL [14,15,18]:

- mysql
  - nástroj pro práci s databází, pracující pod příkazovou řádkou,
  - všestranné možnosti,
  - práce probíhá pomocí unixových příkazů a parametrů v příkazové řádce,
  - všemocný nástroj určený spíše pokročilejším uživatelům.
- phpMyAdmin
  - jednoduchý, kvalitní, velmi aktivně vyvíjen a nejpopulárnější nástroj pro správu databáze přes webové rozhraní,
  - napsaný v jazyku PHP a přeložen do 57 jazyků.
- MySQL Workbench
  - aplikace přímo od výrobce databáze,
  - integrované prostředí obsahující nástroje pro:
    - modelování databáze,
    - SQL příkazy,
    - administraci databáze.

Do budoucna se přesně neví, co ORACLE udělá s MySQL databází (ORACLE totiž svojí databázi má a MySQL byl pro něj určitou konkurencí). Nejbližší projekt projektu MySQL a to je databáze PostgreSQL.

PostgreSQL je Open-Source relační databáze, která má za sebou více než 15 let vývoje. Jedná se o velmi kvalitní databázi, která si zakládá na spolehlivosti a bezpečnosti. Je velmi často srovnávána právě s MySQL a nabízí více vlastností a možností. Naopak MySQL je více rozšířená a má daleko větší komunitu. PostgreSQL je tedy kvalitnější náhradou za MySQL pro případ zastavení vývoje firmou ORACLE. [17]

2.8 HTTP

HTTP (HyperText Transfer Protokol) je bezstavovým aplikačním protokolem, který pracuje na aplikační vrstvě v sadě protokolu TCP/IP, ve které jsou i další protokoly, například: FTP, DHCP, NFS či Telnet. Protokol HTTP spolu s jazykem HTML (blíže popsán níže) a schématem URI (Universal Resource Identifier) vytváří základ pro internetovou službu WWW (World Wide Web). [1]

Samotná komunikace protokolu HTTP je založena na principu požadavek/odpověď. To znamená, že klient (webový prohlížeč) odešle požadavek na server. Server tento požadavek zpracuje a odešle odpověď, která se skládá z hlavičky a požadovaného dokumentu. Pokud klient vzápětí provede stejný požadavek, server neumí poznat, jestli požadavek souvisí s předchozím a tak ho provede nezávisle na předchozím dotazu a to znamená, že je protokol HTTP bezstavový. [1,3]

HTTPS značí bezpečnější verzi protokolu HTTP, která je šifrovaná pomocí SSL. HTTPS používá standardně také jiný port, místo 80 u HTTP používá 443. [3]

Existuje několik metod HTTP protokolu [1]:

- GET- nejpoužívanější metoda vůbec. Metoda posílá požadavek na určitý objekt se zasláním případných dat (například proměnné, session id), která jsou obsažena v URL adrese. Tato metoda má své úskalí v tom, že data jsou lehce viditelná třetím stranám (případný útočník, proxy server) a tak se nehodí na posílání citlivých údajů – například přihlašovací údaje. Navíc jsou data omezena velikostí (max. 256 znaků),
- POST- metoda odesílající data na server. Metoda vhodnější pro posílání velkých a citlivých dat. Data nejdou vidět v URL, ale jsou součástí HTTP dotazu. Nejčastější použití je ve formulářích na webu,
- HEAD- podobná metodě GET, ale nepředává data,
- PUT- nahraje data na server. V praxi se používá FTP nebo bezpečnější SCP/SSH,
- DELETE- smaže uvedený objekt na serveru,
- TRACE- metoda, která spolu s odpovědí serveru pošle i požadavek klienta – možnost ověření neoprávněného zásahu do hlavičky požadavku,
- OPTIONS- metoda, která informuje o podporovaných metodách serveru.

Na příkladu komunikace si vysvětlíme dotaz a odpověď mezi klientem a serverem. První řádek dotazu vypadá ve tvaru: metoda objekt HTTP/verze. Přičemž je dotaz doplněn o další hlavičky, informující například o doménovém jménu serveru, použitém internetovém prohlížeči a také o znakové sadě klienta. [1,3]

Příklad takovéto hlavičky dotazu [3]:

```
GET http://www.ekf.vsb.cz HTTP/1.1
Host: ekf.vsb.cz
User-Agent: Mozilla/5.0 (X11; U; Linux x86_64; cs-CZ; rv:1.9.1.8) Gecko/20100214
Ubuntu/9.10 (karmic) Firefox/3.5.8
Accept-Charset: UTF-8,*
```

Po zpracování dotazu klienta, server odpoví takto [3]:

```
HTTP/1.1 200 OK
Date: Mon, 5 Apr 2010 22:11:45 GMT
Server: Apache/2.2.14 (Unix) PHP/5.3.1
Vary: Accept-Encoding,Cookie
Content-Language: cs
Content-Type: text/html; charset=utf-8

<HTML>
<HEAD></HEAD>
<BODY><H1>Hello World</H1>
</BODY>
</HTML>
```

Takto vypadá odpověď serveru, která se dělí na dvě části: hlavičku a HTML dokument. V hlavičce jsou uvedeny informace: protokol, verze, stavový kód, datum, čas, verze serveru a php, znaková sada a další informace. Po hlavičce následuje prázdný řádek (odděluje hlavičku a dokument) a samotný dokument, který žádal klient, v tomto případě HMTL stránka. [1,3]

2.9 HTML

HTML (HyperText Markup Language) je značkovací jazyk pro vytváření stránek na internetu. Kód HTML je zobrazován uživatelům jako webová stránka v prohlížeči klienta. V roce 1990 byl tento jazyk navržen a vyvíjí se do současnosti. Standardy HTML (ale také HTTP, CSS či URL) stanovuje mezinárodní konsorcium W3C. [11]

Struktura dokumentu HTML [11]:

- Deklarace DTD, která stanovuje definici pravidel. Deklaruje se direktivou *<! DOCTYPE,*

- Element HTML (ohraničený značkami *<HTML></HTML>*) reprezentuje celý dokument,
- Hlavička elementu, obsahující metadata, například : název dokumentu, jazyk, kódování. Je ohraničený značkami *<HEAD></HEAD>,*
- Tělo dokumentu, který obsahuje vlastní text dokumentu. Je ohraničený značkami *<BODY></BODY>*.

V samotném tělu dokumentu BODY dochází k definování HTML stránky dle značek, které jsou určeny standardy. Samotné tělo se skládá ze značek a jejich atributů, které se uzavírají do hranatých závorek (< >). Tyto značky jsou většinou párové (důležité pro správné zobrazení), ale existují i nepárové. Párové značky obsahují počáteční a koncové značky, které mají stejné názvy, avšak koncová značka má před názvem lomítko (/). Mezi dvojicí značek je umístěn obsah- text, který formátován dle značek, v nichž je uzavřen. [11]

Dvojice značek a text mezi nimi tvoří element dokumentu. Součást obsahu elementu může být další element. Důležité také je, aby vnořený element skončil (koncovou značkou) dříve než obklopující element (nesmí dojít ke křížení dvojicí značek). [11]

Příklad dokumentu HTML [11]:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
  <!-- toto je komentář -->
  <head>
    <title>Titulek stránky</title>
  </head>
  <!-- tělo dokumentu -->
  <body>
    <h1>Nadpis stránky</h1>
    <p><i>Toto je tělo dokumentu</i></p>
  </body>
</html>
```

Od stylistických značek, které určují vzhled elementu (barva, styl písma), se v současnosti upouští. Nahradily je kaskádové styly, které jsou více popsány v další kapitole. [11]

2.10 CSS

HTML se stará o strukturu a obsah textů na www stránkách. Samozřejmě v HTML

lze definovat i formu textu (barvy, velikost písma, pozadí), ale na tomto poli ho vystřídal novější a dnes už primárně využívaný způsob formátování textu- tzv. CSS styly. Návrhářům stránek tímto došlo k oddělení zpracování textu a struktury pomocí HTML a formátování textu pomocí CSS stylů. [9]

CSS (Cascading Style Sheets neboli kaskádové styly) je značkovací jazyk sloužící pro grafickou úpravu www stránek. Tento jazyk byl opět navržen a je i spravován organizací W3C. [9]

Existují tři způsoby použití CSS stylů do HTML stránky [9]:

- Zápis přímo do konkrétního elementu HTML jako atribut *style*
  - Tento způsob se využívá málo, protože edituje jen patřičný element HTML kódu. Oproti HTML formátování elementu představuje výhodu jen v širších možnostech CSS stylů.
- Zápis pomocí elementu *style*
  - Obsahuje seznam stylů, které se aplikují na celý HTML dokument. Seznam těchto stylů je definovaný mezi značkami <*style*></*style*>.
- Přilinkování externího CSS souboru
  - Tento způsob je nejpoužívanější. Má nesmírnou výhodu v tom, že jeden soubor CSS definuje formátování elementů ve všech HTML stránkách, kde je vložen *link* element odkazující na CSS soubor.

Jak jsem zmínil, nejčastěji je použití CSS stylů v externím souboru. Tak pro tento případ si uvedeme jednoduchý příklad. Nejprve je potřeba vložit tento řádek do hlavičky HTML dokumentu [9]:

```
<link rel="stylesheet" type="text/css" href="styly.css">
```

Poté nám nic nebrání ve vytvoření nového CSS souboru *styly.css*, který může například vypadat takto [9]:

```
body {
  background-color: white;
  color: black;
  padding: 10px!Important;
}
```

Při tvorbě webové grafiky je tedy práce s CSS styly důležitá a při vytváření stránek hojně využívána, a to pro svoji jednoduchost nastavování elementů ve všech našich stránkách HTML z jednoho místa- souboru CSS. [9]

2.11 PHP

Při tvorbě našeho systému je potřeba zajistit dynamiku na straně serveru. Potřebujeme vytvářet HTML stránky dynamicky dle algoritmu a s možností přístupu k databázi. Pro zajištění dynamiky na straně serveru je nutné použít kvalitní skriptovací jazyk s dobrou podporou MySQL. Zvolil jsem jazyk PHP- poslední složka LAMP serveru. [1,3,4]

PHP (Hypertext Preprocessor) je výkonný skriptovací jazyk pro programování dynamických internetových stránek na straně serveru. K správnému zobrazení stránek je zapotřebí mít na serveru nainstalovaný webový server Apache a samotné PHP. Důležité je také (ať o tom server ví, že zpracovává PHP) vytvářet soubory s koncovkou *.php. [3,24]

Jazyk se začleňuje přímo do struktury HTML pomocí vsuvek, které jsou programově zpracovány serverem. Uživateli je poté poslán výsledek ve formě čistého HTML dokumentu.

Mezi základní vlastnosti PHP patří [3,24]:

- multiplatformnost,
- jednoduchá syntaxe,
- Open-Source software,
- podpora více než 20 databázových systémů (např. MySQL, SQLite, PostgreSQL, a mnoho dalších),
- PHP je součástí „LAMP“,
- vysoká podpora ze strany web hostingů,
- PHP má výborný manuál (http://www.php.net),
- velká rychlost učení jazyka.

V jazyku PHP je napsáno mnoho hotových projektů:

- phpBB (diskuzní fóra),
- WordPress (redakční systém),
- phpMyAdmin/phpPgAdmin (aplikace na správu MySQL/PostgreSQL databáze),
- Joomla! (redakční systém).

Na konec menší představení jazyka:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

2.12 JavaScript

Při tvorbě jsem potřeboval zajistit spouštění skriptů na straně klienta, takže jsem použil i skriptovací jazyk JavaScript, což je multiplatformní skriptovací jazyk používaný při vytváření dynamických stránek, který se do HTML kódu vkládá podobně jako PHP, konkrétně mezi vsuvky *<script></script>*. Tento kód je webovým serverem přehlížen a je posílán v HTML dokumentu uživateli (zdrojový kód JavaScriptu může tedy klient přečíst), který jej spustí ve svém prohlížeči. [10]

Skript zpracovává klient a server se zpracováním nemusí zabývat. Ale přináší to také své neduhy, například zakázáním JavaScriptu v www prohlížeči klienta způsobí nefunkčnost skriptu. JavaScript také z bezpečnostních důvodů neumí přistupovat k souborům uživatele, ale také ani ukládat soubory. [10]

Ukázka JavaScript kódu [10]:

```
<html>
<head>
<title>Příklad na metodu prompt()</title>
</head>

<body>
<script>
x = prompt("Zadej svoje jméno", "");

document.write("Tvoje jméno tedy je ");
document.write(x);
</script>


</body>
</html>
```

3 Analýza současného stavu

Zatím neexistuje informační systém na ekonomické fakultě VŠB, kde by se člověk dostal k impaktovaným příspěvkům a případně tyto příspěvky citoval. Je také zapotřebí citovat se navzájem ve skupině tak, aby bylo dosaženo vyššího hodnocení vědecké práce. Uživatelům budoucího systému chybělo hlavně sdílení příspěvků. Chybějící informační systém, který by poskytoval rychlý a jednoduchý přehled o impaktovaných příspěvcích, bylo potřeba vytvořit minimálně v jednoduché podobě, který ale bude muset být přístupný odkudkoliv. Proto byl nakonec vytvořen internetový informační systém, s jednoduchým a přehledným vzhledem, umístěný na webovým portále a tak plnící jeden z požadavků zadavatele - dostupnost systému.

Systém bude sloužit uživatelům v rámci ČSSI skupiny, ale v blízké budoucnosti lze očekávat rozšíření služeb informačního systému i pro ostatní fakulty či vysoké školy, které by mohly také přispívat a publikovat příspěvky s impaktovaným faktorem.

Současný stav na internetu nahrává myslet při vývoji systému i na jednu důležitou věc - bezpečnost. Přístupnost systému odkudkoliv je věc dobrá a pro využití tohoto systému potřebná, ale přináší to jisté úskalí, na které je třeba myslet při vývoji systému. Je třeba myslet na bezpečnost. Dostupnost systému nemůže být na úkor bezpečnosti. Je třeba chránit citace a osobní údaje registrovaných uživatelů před kolizemi a hlavně útoky, kterých v dnešní době je hojně.

Proto při vývoji na projektu byla provedena bezpečnostní analýza, analyzující nejen současný stav hrozeb, ale i předpověď vývoje hrozeb v brzké budoucnosti. Při vývoji bylo zapotřebí využít nejnovější dostupné verze softwarových produktů, které byly na bezpečnost dost připraveny. Zároveň byla stanovena bezpečnostní politika celého systému a použité bezpečnostní algoritmy.

Součástí analýzy současného stavu je analýza pomocí USE CASE diagramů, která zachycuje chování systému za různých situací ve formě reakcí systému na požadavky primárního aktéra – uživatele systému.

3.1 Návrh použití systému pomocí USE CASE

3.1.1 Analýza slovního obsahu příběhu

Uživatel chce sdílet a citovat příspěvky. Musí otevřít webový prohlížeč. Uživatel musí zadat webovou adresu systému do svého prohlížeče. Uživatel se nejprve musí zaregistrovat do systému. Uživatel čeká na povolení moderátorem. Uživatel se přihlásí do systému. Uživatel si vybere kategorii příspěvku a vyplní informace o příspěvku. Uživatel vloží nový příspěvek. Uživatel prohlíží všechny příspěvky a cituje příspěvky jiných uživatelů systému. Uživatel se odhlašuje ze systému.

3.1.2 Tabulka kandidátů na USE CASE

Aktéři – podstatná jména	Kandidát Use Case - slovesa
Uživatel	Chce sdílet a citovat příspěvky Otevře internetový prohlížeč Zadá webovou adresu Registruje do systému Čeká na povolení moderátorem Přihlašuje se do systému Vybírá kategorii příspěvku Vyplňuje informace o příspěvku Vkládá nový příspěvek Prohlíží příspěvky Cituje příspěvky jiných uživatelů Odhlásí se ze systému
Moderátor	Dohlíží na chod systému Povoluje registrace uživatelů
Systém	Registruje nové uživatele Ukládá příspěvky Zobrazuje příspěvky Evidence uživatelů
Administrátor	Dohlíží na chod systému Řeší technické problémy

3.1.3 Vstupně – výstupní seznam

Téma	Vstup	Výstup
Registrace do systému	vstup	
Povolení registrace	vstup	
Přihlášení do systému	vstup	
Vkládání příspěvku	vstup	
Prohlížení příspěvků		výstup
Odhlášení ze systému		výstup

3.1.4 katalog uživatelských požadavků

Aktér	Požadavek (cíl)	Priorita
Uživatel	Mít potřebu citovat a sdílet své citace	1
	Citovat jiné uživatele	2
Moderátor	Povoluje vstup do systému	2
Administrátor systému	Stará se o chod systému (bezpečnost, problémy)	1
	Stará se o vývoj IS	2
Systém	Eviduje příspěvky a uživatele	1

3.1.5 Tabulka záměrů případů užití

Aktér	Požadavek (cíl)	Záměr
Uživatel	Potřebuje sdílet své citace	Registrace do systému
	Citování ostatních uživatelů a vkládání svých citací	Přihlášení do systému a použít jej
Moderátor	Mít přehled o systému a povolovat nové uživatele	Dohlíží na chod systému
Administrátor systému	Stará se o bezproblémový chod systém	Poskytuje podporu uživatelům při používání systému

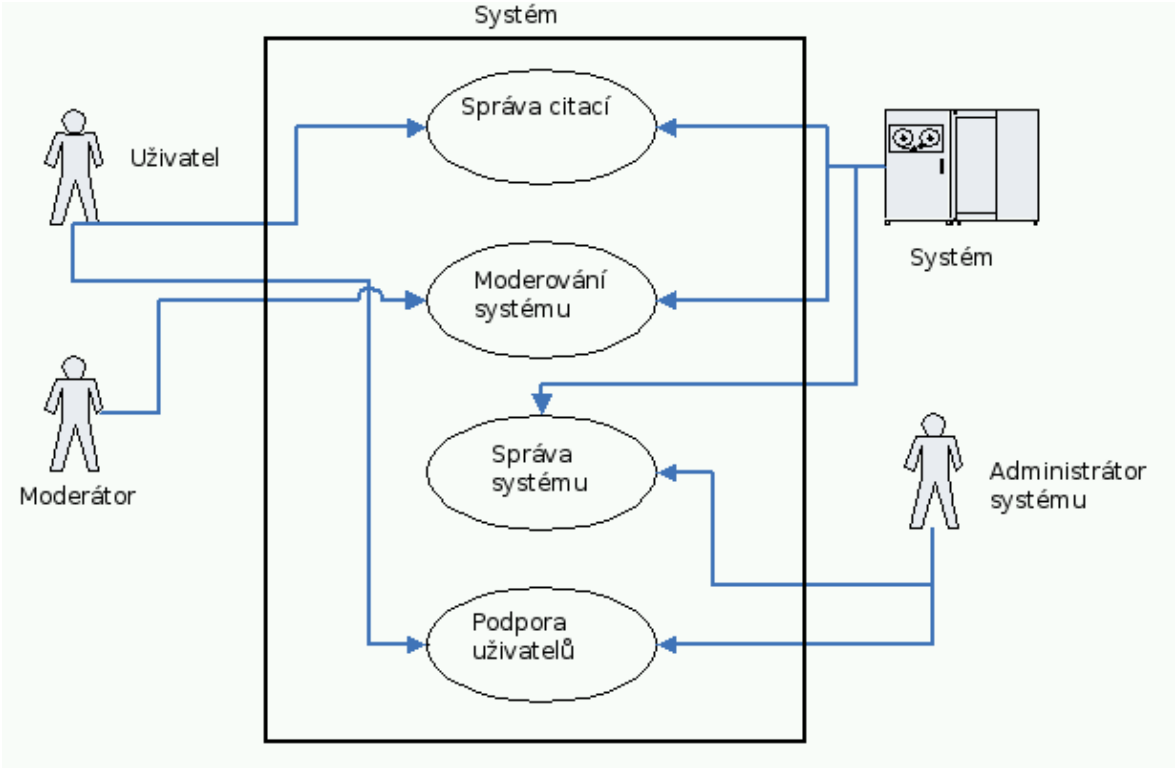
3.1.6 Tabulka identifikace a specifikace aktéra

	Účastník	Primární aktér	Pomocný aktér	Vyvíjený systém
Uživatel		x		
Moderátor	x		x	
Administrátor systému			x	
Systém				x

3.1.7 Tabulka profilu aktér

Jméno aktéra	Profil: Kvalifikace a schopnosti
Uživatel	Jakákoliv osoba, která se zaregistruje v systému a je povolena moderátorem. Využívá systém pro citování.
Moderátor	Osoba kontrolující chod systému.
Administrátor systému	Tvůrce systému, který systém doplňuje o další funkce a stará se o bezpečnost a stabilitu systému.
Systém	Má schopnost spravovat uživatele a příspěvky. Kontroluje správnost zadávaných citací a zobrazuje je ostatním uživatelům.

3.1.8 Schéma USE CASE



Obrázek 3.1: Schéma USE CASE

3.1.9 Hlavní scénář

Identifikace: UC1

Název: Vložení příspěvku

Primární aktér: Uživatel

Rozsah: Vložení příspěvku

Úroveň: uživatelský cíl

Účastníci a zájmy:

Uživatel – vkládá příspěvek

Moderátor – kontroluje chod systému

Administrátor systému - vede systém v provozuschopném a aktuálním stavu

Systém – spravuje uživatelské konta a příspěvky

Vstupní podmínky:

- Uživatel je řádně zaregistrovaný v systému.
- Systém je v provozuschopném stavu.

Minimální záruky:

Server zjistí chybu a dotáže se uživatele na doplňují informace.

Záruky úspěchu:

Uživatel úspěšně vloží příspěvek.

Spouštěč: Uživatel chce vložit příspěvek do systému.

Hlavní úspěšný scénář:

- Uživatel zadá přihlašovací údaje do přihlašovacího formuláře systému a přihlásí se.
- V pravém menu systému vybere uživatel možnost „Přidat příspěvek“ a poté typ vkládaného příspěvku.
- Uživatel vyplní potřebné informace o příspěvku a případně vloží příspěvek v souboru PDF.
- Uživatel potvrdí vkládaný příspěvek.
- Uživatel vidí a zkontroluje svůj vložený příspěvek v systému, splňující normu ČSN 690.
- Uživatel se odhlásí ze systému.

Rozšíření:

1a: Uživatel se nepřihlásí do systému.

1a1 Uživatel zadal nesprávné přihlašovací údaje.

1a2: Uživatel zadá nové přihlašovací údaje.

2a: Uživatel nevidí svůj požadovaný typ vkládaného příspěvku.

2a1: Uživatel kontaktuje administrátora systému.

2a2: Případ užití končí.

3a: Uživatel nemůže vložit soubor s jinou příponou než PDF.

3a1: Uživatel kontaktuje administrátora systému.

3a2: Uživatel musí vložit příspěvek v souboru s příponou PDF.

4a: Systém zobrazí chybové hlášení uživateli.

4a1: Uživatel musí upravit příspěvek dle chybového hlášení.

4b: Systém vypíše chybu o překročení velikosti souboru nebo neplatné příponě.

4b1: Uživatel musí vložit soubor o maximální velikosti 20 MB a s příponou PDF.

5a: Uživatel zapomněl či špatně zadal položku citace.

5a1: Uživatel upraví již vložený příspěvek.

4 Rozšíření informačních služeb

Kapitolu jsem rozdělil do podkapitol, ve kterých postupně popíšu jaké softwarové nástroje jsem použil, jak jsem pracoval s PHP, MySQL databází a jaké bezpečnostní algoritmy, které byly v systému použity.

4.1 Příprava k práci

Pří vývoji informačního systému jsem použil následující software a manuály.

Použitý software:

- NetBeans IDE 6.7.1 – vývojové prostředí určené pro programování a vývoj primárně Java aplikací, ale nabízí i podporu dalších jazyků: C/C++, Ruby, HTML, PHP,
- phpMyAdmin – prostředí (napsané v jazyku PHP) pro správu MySQL databáze,
- FileZilla 3.3.1 – klient pro práci s protokoly FTP, SFTP a FTPS,
- Internet Explorer 8, Firefox 3.5.7 – webové prohlížeče pro zobrazení HTML výstupu.

Použité manuály:

- http://www.php.net,
- http://dev.mysql.com/doc/,
- http://httpd.apache.org/docs/.

Primárně bylo potřeba vývojové prostředí NetBeans a oficiální manuál k jazyku PHP. Zároveň (ačkoliv to nebylo nutné, neboť správa serveru spadala pod správce serveru) jsem se naučil i administraci linuxového serveru, konfiguraci PHP a Apache serveru. Ale prioritou bylo vytvořit kvalitní kód PHP a ten nadále spravovat a upravovat dle potřeb uživatelů. Samozřejmě jsem si v domácích podmínkách správu serveru vyzkoušel a doplnil tak vědomosti.

4.2 Databázový model

Nejdříve bylo zapotřebí v prostředí phpMyAdmin vytvořit databázi *cssimorava*, která byla doplněna o 4 tabulky: *tblautor*, *tbldiskuze*, *tblprispevky* a *tblkonference*.

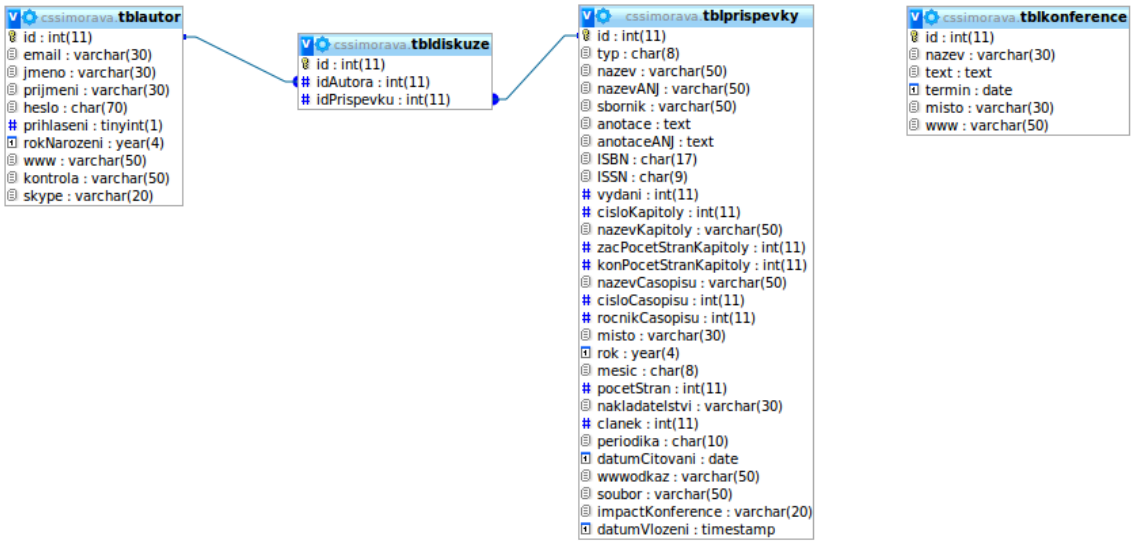
Pro všechny tabulky byl použit způsob uložení MyISAM (výchozí formát MySQL databáze), které pro toto použití bylo ideální. Celá databáze je kódována ve znakové sadě UTF-8. Stejné kódování je použito i na HTML stránkách.

V databázi jsou použity tyto datové typy:

- tinyint – pro velmi malá čísla v rozsahu [-128; 127],
- int – celá čísla v rozsahu [-2147483648; 2147483647],
- varchar – textový řetězec s proměnnou délkou (počet znaků),
- text – datový typ sloužící pro uložení dlouhých testů,
- char – textový řetězec s pevně stanovenou šířkou,
- year – typ pro uložení roku,
- date – datový typ pro uložení data,
- timestamp – datový typ ukládající aktuální datum při SQL příkazech (INSERT a UPDATE).

4.2.1 Datový model

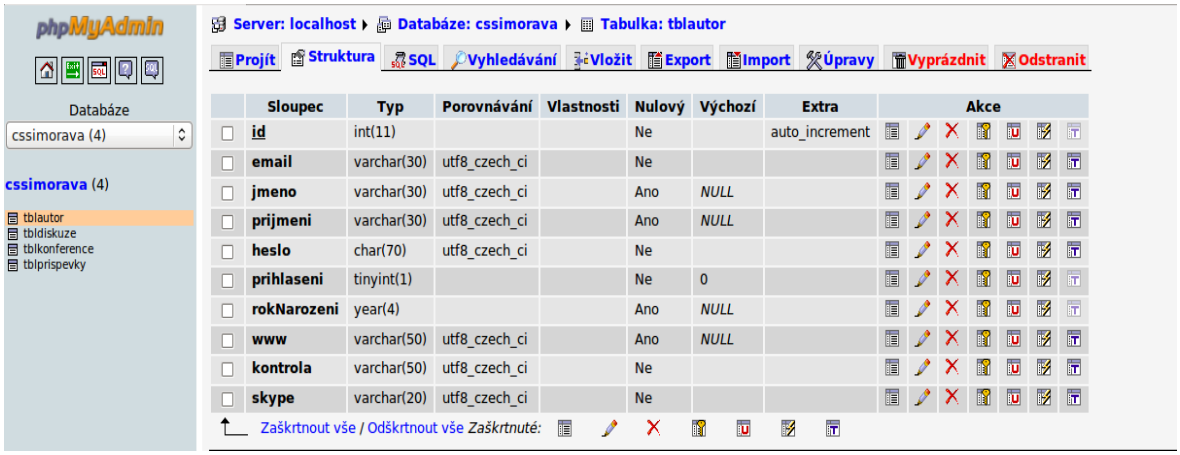
V systému byly vytvořeny 4 tabulky, přičemž 3 z nich byly spojeny relačními vztahy (1:N), které spojují primární a cizí klíče databázových tabulek. Díky kombinaci dvou relačních vztahů 1:N modelujeme vztah M:N, který požadujeme v datovém modelu.



Obrázek 4.1: Datový model vytvořený aplikací phpMyAdmin

4.2.2 Tabulka tblautor

Tato tabulka obsahuje informace o registrovaných autorech citací (povolených i nepovolených administrátorem) a obsahuje i spoluautory citací (spoluautor citace, který není v systému zaregistrovaný, ale je použit v nějaké citaci).



Obrázek 4.2: Tabulka tblautor

Takže systém eviduje 3 druhy uživatelů: povolený registrovaný, nepovolený registrovaný a nakonec spoluautor citace použitý v nějaké citaci.

Abychom tyto uživatele odlišili musíme spolu s údaji o autorech zavést další sloupec *prihlaseni*, který podle hodnoty (0,1 nebo 2) rozliší do jaké skupiny autorů patří.

Na obrázku lze vidět prostředí aplikace phpMyAdmin a v něm struktura tabulky *tblautor* i se všemi použitými sloupci a jejich datovými typy či vlastnostmi.

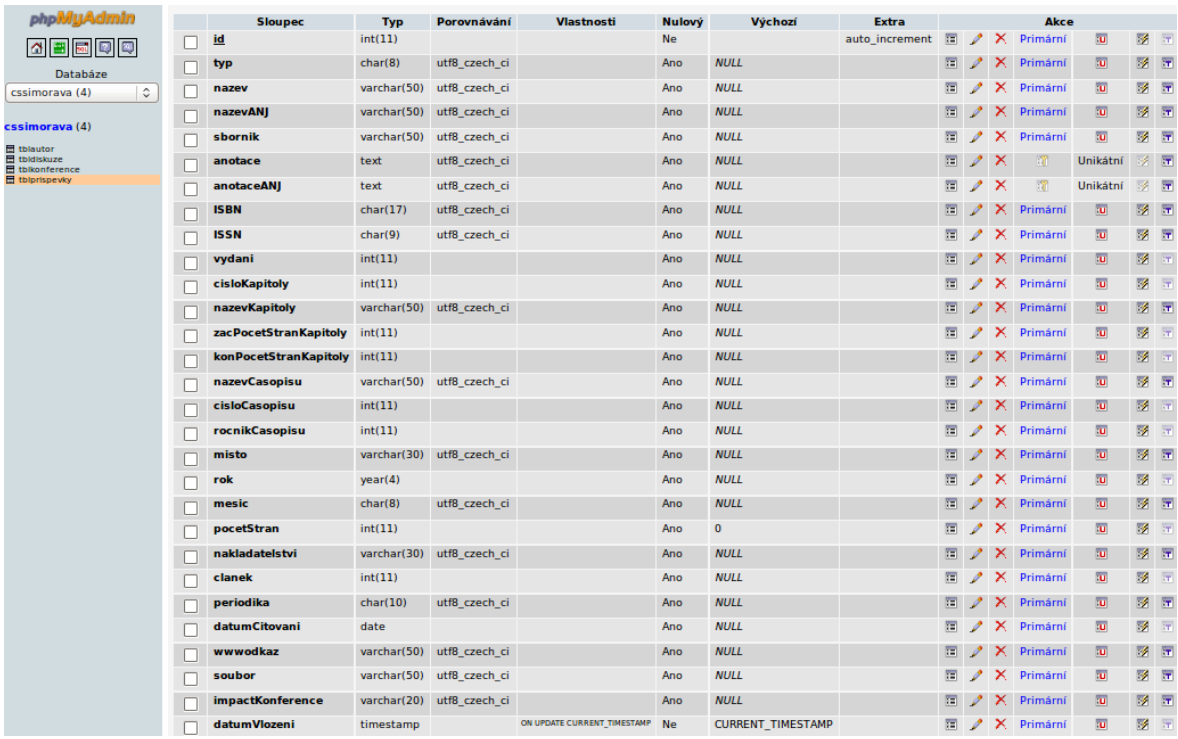
Použité sloupce, jejich datové typy a vlastnosti tabulky *tblautor* vidíte v obrázku. Ale doplnil bych pár poznámek k této tabulce:

- id – (PK- Primary key- Primární klíč) jednoznačný číselný identifikátor pro autora,
- heslo – heslo nemůže být uložené v nešifrované formě jako text vkládaný do přihlašovacího formuláře, protože to nese značné bezpečnostní riziko. Heslo zaregistrovaného uživatele je šifrováno šifrovací metodou SHA-1,
- kontrola – je email šifrován metodou SHA-1.

4.2.3 Tabulka tblprispevky

V této tabulce jsou informace o jednotlivých citacích. V následujícím obrázku lze

vidět všechny použité sloupce různých datových typů.



Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
id	int(11)			Ne		auto_increment	Primární
typ	char(8)	utf8_czech_ci		Ano	NULL		Primární
nazev	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
nazevANJ	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
sbornik	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
anotace	text	utf8_czech_ci		Ano	NULL		Unikátní
anotaceANJ	text	utf8_czech_ci		Ano	NULL		Unikátní
ISBN	char(17)	utf8_czech_ci		Ano	NULL		Primární
ISSN	char(9)	utf8_czech_ci		Ano	NULL		Primární
vydani	int(11)			Ano	NULL		Primární
cisloKapitoly	int(11)			Ano	NULL		Primární
nazevKapitoly	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
zacPocetStranKapitoly	int(11)			Ano	NULL		Primární
konPocetStranKapitoly	int(11)			Ano	NULL		Primární
nazevCasopisu	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
cisloCasopisu	int(11)			Ano	NULL		Primární
rocnikCasopisu	int(11)			Ano	NULL		Primární
misto	varchar(30)	utf8_czech_ci		Ano	NULL		Primární
rok	year(4)			Ano	NULL		Primární
mesic	char(8)	utf8_czech_ci		Ano	NULL		Primární
pocetStran	int(11)			Ano	0		Primární
nakladatelstvi	varchar(30)	utf8_czech_ci		Ano	NULL		Primární
clanek	int(11)			Ano	NULL		Primární
periodika	char(10)	utf8_czech_ci		Ano	NULL		Primární
datumCitovani	date			Ano	NULL		Primární
wwwodkaz	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
soubor	varchar(50)	utf8_czech_ci		Ano	NULL		Primární
impactKonference	varchar(20)	utf8_czech_ci		Ano	NULL		Primární
datumVlozeni	timestamp		ON UPDATE CURRENT_TIMESTAMP	Ne	CURRENT_TIMESTAMP		Primární

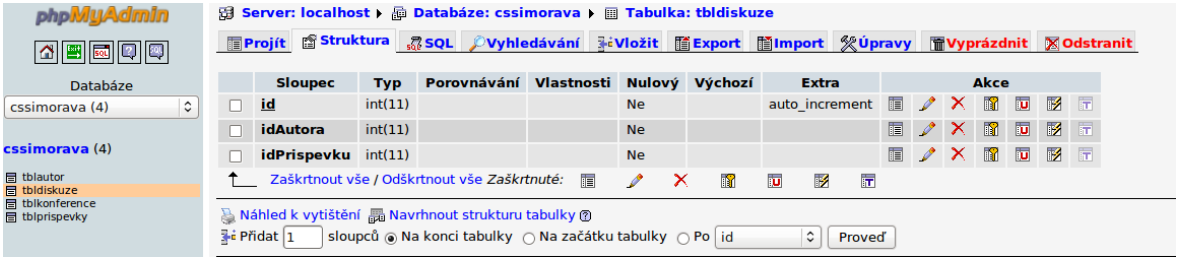
Obrázek 4.3: Tabulka tblprispevky

4.2.4 Tabulka tbldiskuze

V předchozích tabulkách jsme měli údaje o autorech a příspěvcích. Teď nám schází tabulka, která tyto údaje spojí. Tabulka *tbldiskuze* slouží k propojení tabulek *tblautor* a *tblprispevky*. Díky této tabulce lze přiřadit více autorů k jedné citaci a více citací k jednomu autoru.

V databázích nelze vytvořit vztah M:N (přiřazení více záznamů z tabulky jedné k více záznamů tabulky druhé), i když tento vztah požadujeme. Musíme si pomoct vztahy 1:M a 1:N (jeden záznam odpovídá více záznamům tabulky druhé) a tím musíme vytvořit tabulku třetí- pomocnou, v našem případě tabulku *tbldiskuze*. Tabulka *tbldiskuze* se skládá z primárního klíče *id*, sloupce *idAutora* a *idPrispevku*.

Takže pomocí jedinečného sloupce id dostaneme jedinečné spojení autora a samotného příspěvku v databázi. Když má například autor více příspěvků, bude v tabulce *tbldiskuze* více záznamů se stejným *idAutora* a rozdílnými *idPrispevku*.



Obrázek 4.4: Tabulka tbldiskuze

4.2.5 Tabulka tblkonference

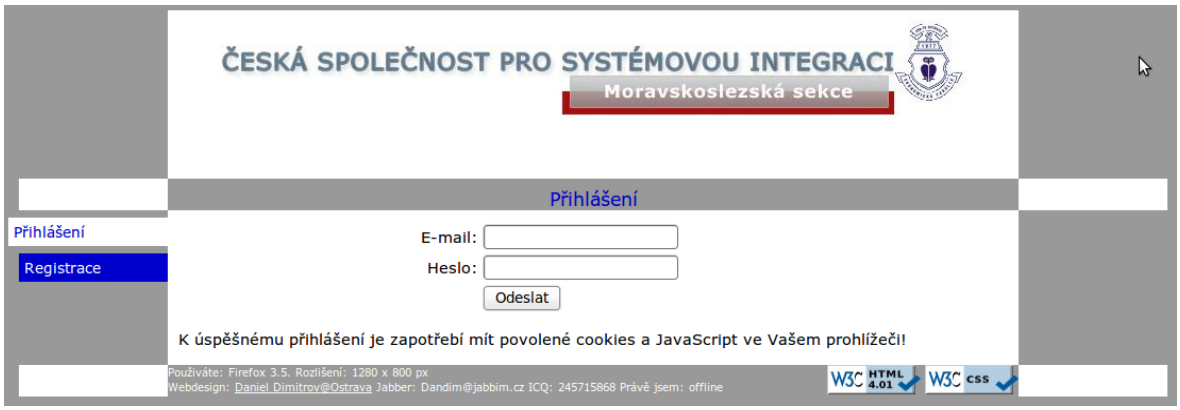
Poslední tabulkou je zároveň i nejméně důležitou pro samotný chod systému. Jde o doplňující tabulku poskytující informace o připravovaných konferencí týkající se skupiny ČSSI, případně i dalších z oboru IT.

Obrázek 4.5: Tabulka tblkonference

4.3 PHP kód

Tato kapitola popisuje zpracování jednotlivých problémů při implementaci systému, fungování systému a ukázky grafického uživatelského prostředí či samotného programovacího kódu v jazyku PHP. Kapitola je rozdělena dle funkcionalit, které systém nabízí uživatelům systému. Dle konkrétní funkce bude přizpůsoben i obsah podkapitoly tak, aby byla podkapitola vhodně vysvětlena. Takže na začátek této kapitoly by bylo vhodné ukázat, to co uživatel systému uvidí prvně po zadání webové adresy systému do webového prohlížeče, tedy úvodní webovou stránku. Úvodní webová stránka obsahuje menu o dvou položkách: Přihlášení a Registrace.

4.3.1 Přihlášení



Obrázek 4.6: Úvodní stránka

Jde o velmi jednoduchou úvodní stránku, jejíž hlavní funkcionalitou je přihlášení registrovaných uživatelů, registrace nových uživatelů a nakonec uvedení pár informací v patičce stránky. Tato patička se objevuje v celém systému a při vytváření patičky byl kladen důraz na informovanost uživatelů tak, aby uživatel měl vždy na očích kontakt na administrátora a při sebemenších problému jej použil. Takže v této patičce jsou uvedeny kontaktní informace na administrátora systému (e-mail, jabber, ICQ), použitý webový prohlížeč, rozlišení obrazovky uživatele a validace HTML či CSS dle skupiny W3C.

Přes tuto úvodní stránku se uživatelé přihlašují pomocí zaregistrovaných e-mailů a svých hesel. Pro pochopení problematiky si ukážeme script přihlašovacího formuláře.

```
<FORM action="index.php?odkaz=prihlaseni" method="POST">
   <TABLE align="center" width="400">
      <TR><TD align="right">E-mail:</TD>
                  <TD align="left"><INPUT type="text" name="mail" size="20" value="<?
if(isset($_POST["mail"])) echo $_POST["mail"]?>"></TD></TR>
      <TR><TD align="right">Heslo:</TD>
               <TD align="left"><INPUT type="password" name="heslo" size="20" ><?echo (!
empty($chyba["jmeno"]))?"<DIV id=\"error\">Neplatné uživatelské údaje!</DIV>":"";?
></TD></TR>
                              <TR><TD></TD><TD align="left"><INPUT type="submit"
value="Odeslat"></TD></TR>
   </TABLE>
      </FORM>
<p>K úspěšnému přihlášení je zapotřebí mít povolené cookies a JavaScript ve Vašem prohlížeči!
</p>
```

Tento formulář posílá přihlašovací údaje pomocí metody POST „sám sobě“, tedy na stejnou webovou stránku, kde se nachází již uvedený formulář. Takže na začátku této webové stránky musí být umístěn script PHP, který ověří zadané přihlašovací údaje.

Formulář také umožňuje vypsání e-mailu do vstupního pole *mail*, pokud

se proměnná mail nachází v HTML požadavku. Takže tato proměnná se v požadavku nachází odesláním formuláře. K této situaci dojde, když uživatel vyplní nesprávné údaje. A proto, aby nemusel opět vyplňovat pole E-mail, je už toto pole vyplněné již hodnotou, kterou uživatel zadal a poslal pomocí formuláře.

Toto doplňování samozřejmě neplatí pro heslo. To musí uživatel zadávat vždy. Teď se dostaneme ke zpracování požadavku PHP scriptem umístěným na začátku HTML stránky.

```
<?php
include 'funkce.php';
$BudemeZobrazovat=true;
if(isset($_SESSION["id"])){
        $BudemeZobrazovat=false;
        session_regenerate_id();
         header('Location: ./login.php?odkaz=hlavni');}
      if(!empty($_POST["mail"])){
          $link=mysql_connect(SERVER,UZIVATEL,HESLO) or die("Nelze se připojit k MySQL: " .
mysql_error());
   mysql_select_db(DATABAZE) or die("Nelze vybrat databazi: ". mysql_error());
   mysql_query("SET NAMES 'utf8'");
   foreach ($_POST as $key => $value) {
   $_POST[$key] = mysql_real_escape_string($value);
   $_POST[$key]=htmlspecialchars($value,ENT_QUOTES,'UTF-8');
   }
        $vysledek=mysql_query("select email,heslo,prihlaseni,kontrola from tblautor where
prihlaseni=1 and email='".$_POST['mail']."' and heslo='".sha1($_POST["heslo"])."'");
   if ($vysledek==false) die("Nelze vykonat dotaz: " . mysql_error());
   if (mySQL_num_rows($vysledek)==1){
      session_start();
      if(!isset($_SESSION["id"])){
         $zaznam=MySQL_Fetch_Array($vysledek);
         session_regenerate_id();
         $_SESSION["id"]=$zaznam["kontrola"];
         $BudemeZobrazovat=false;
         header('Location: ./login.php?odkaz=hlavni');
      }
   }else{
       $chyba["jmeno"]=true;
      }mysql_close($link);}
?>
```

Pomocí tohoto scriptu PHP dochází k přihlašování uživatelů do systému. Jak jsem uvedl je tento script na začátku úvodní stránky HTML, takže tímto kódem prochází každý návštěvník stránek. Pomocí funkce *include* dojde k vložení souboru funkce.php, kde jsou uloženy přihlašovací údaje do databáze MySQL (dále script používá konstanty SERVER, DATABAZE, UZIVATEL a HESLO právě ze souboru funkce.php). Poté dojde k nastavení proměnné *BudemeZobrazovat* na hodnotu TRUE, což zapřičiní zobrazení přihlašovacího formuláře.

Je zapotřebí ověřit, zda není nastavena SESSION daného uživatele (např.

nesprávným odhlášením). Jestli je nastavena SESSION dochází k přesměrování na hlavní stránku systému.

Následuje podmínka ověřující prázdnost prvku pole $_POST[“mail“]. Asociativní pole $_POST je naplněno prvky z HTTP požadavku posláno z přihlašovacího formuláře. Takže jestli tato podmínka platí, jsou v poli $_POST uloženy prvky s indexy [“mail“;“heslo“].

V kladné podmínce je navázána databázová konektivita, prvky pole $_POST jsou zkontrolovány na nebezpečné znaky (o bezpečnostní politice bude jednat pozdější kapitola) a poslán dotaz SQL, ve kterém je porovnáván e-mail a heslo. Heslo je ale nejprve zašifrováno metodou SHA1 a poté porovnáváno s hashem v databázi.

Přihlašovací údaje jsou ověřeny v databázi, a pokud jsou správné je nastavena SESSION proměnná konkrétního uživatele (systém musí rozlišit uživatele mezi sebou). Do této proměnné, která je uložena na straně serveru a klientovi je posílán jen identifikátor SESSION pomocí cookies, je uložena hodnota uživatelova e-mailu zašifrována pomocí metody SHA1.

Nakonec dojde k přesměrování na stránku login.php, kde systém ověří uživatele dle SESSION a je umožněn vstup do systému a práce s ním.

4.3.2 Registrace

Pro úspěšnou registraci je zapotřebí vyplnit formulář, uvedený pod tímto textem.

Obrázek 4.7: Registrace nového uživatele

Veškerá pole formuláře jsou kontrolována dle tohoto schématu:

- E-mail je kontrolován na správný tvar e-mailové adresy a na existenci e-mailu v databázi,
- Pole Jméno a Příjmení musí obsahovat minimálně 3 a maximálně 20 znaků. Rovněž je kontrováno, zda uživatel (jméno a příjmení) není už v databázi obsažen,
- Zadaná webová stránka je zkontrolována na záležitosti správné webové stránky. Je zapotřebí zadat adresu ve tvaru: http://www.mojeadresa.cz/,
- Pro rok narození je nastaven interval 1930 až po aktuální rok,
- Spype je jediné nepovinné pole a je omezen jen počtem znaků (20 znaků),
- Heslo je kontrolováno pro svou délku (8-20 znaků) a je nutno jej zadat dvakrát,
- Nakonec je zapotřebí zadat číslo z náhodného obrázku. Jde o bezpečnostní prvek formuláře (rozlišení uživatelů od robotů).

Po vyplnění formuláře dle schématu je formulář poslán na server, kde dojde k naplnění databáze. Tímto ale není proces registrace plně dokončen, protože, jak je uvedeno na začátku práce, je registrace schvalovaná administrátorem.

Při vyplnění formuláře je poslán administrátorovi e-mail, který ověří registrovaného uživatele a rozhodne se ho povolit či ne. Do povolení uživatele administrátorem není registrovaný uživatel oprávněn se do systému přihlásit. Po případném povolení registrace je registrovanému uživateli automaticky poslán e-mail s informací o možnosti přihlášení.

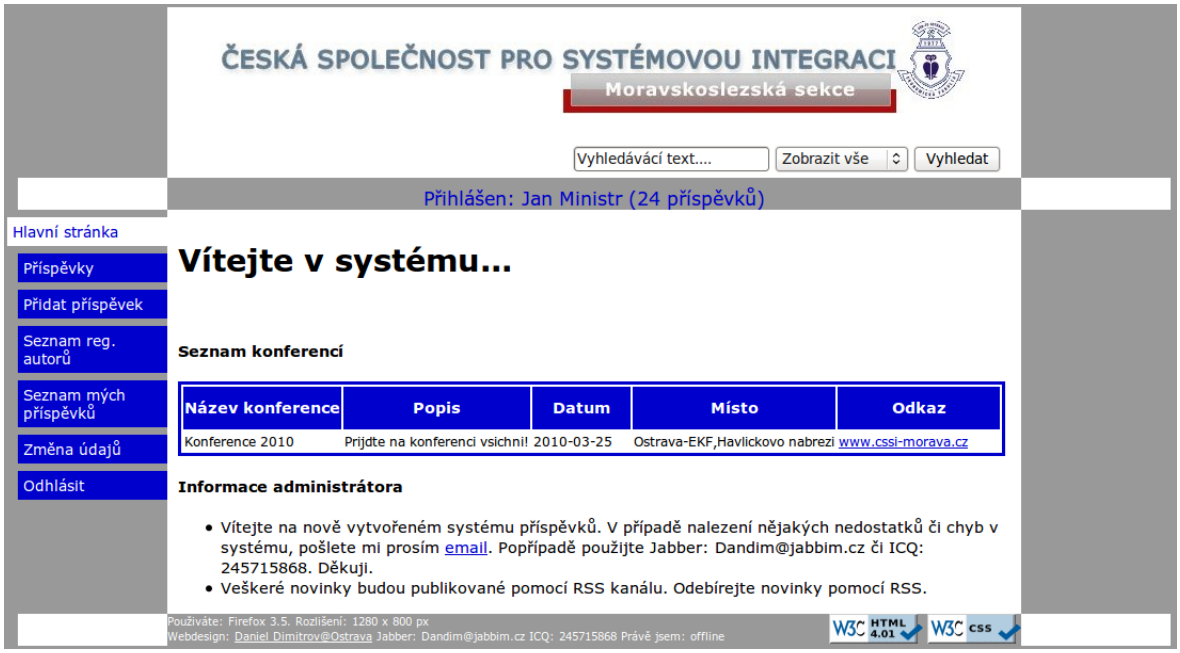
Tímto je zaručena plná kontrola nad registrovanými uživateli a vstup povolen jen těm, kterým je systém určen.

4.3.3 Hlavní strana systému

Po povolené registraci a úspěšnému přihlášení může práce se systémem začít. Jako prvně si představme hlavní stranu systému a elementy zobrazující se na každé stránce systému.

Mezi tyto elementy patří samotné menu, umístěné vlevo, již probranou patičku a hlavičku stránky, na které se nachází vyhledávací modul a informační lišta. Vyhledávací modul umí vyhledat zadaný text v příspěvcích. Je potřeba vybrat kategorii, kde má systém hledat (např. Jméno, příjmení, ISBN, anotace), a poté kliknout na Vyhledat. Informační

lišta nabízí informace o jménu a příjmení přihlášeného uživatele a počtu vložených příspěvků.



Obrázek 4.8: Hlavní stránka systému

Obsah hlavní stránky obsahuje informační tabulku o seznamu konferencí, které se chystají uskutečnit. Tabulka čerpá informace z tabulky *tblkonference* a zobrazuje je ty záznamy, které mají datum větší nebo rovno aktuálnímu datu. Takže, i když jsou v tabulce uloženy záznamy staršího data, v tabulce Seznam konferencí se nezobrazí.

Obsah stránky je doplněn o Informace administrátora, který obsahuje souhrn nejdůležitějších informací pro uživatele. Pro kompletní souhrn změn je zapotřebí odebírat RSS kanál systému. Průběžně je doplňován o případných změnách v systému či vylepšení.

4.3.4 Příspěvky

Na stránce Příspěvky se nacházejí samotné příspěvky citací všech uživatelů a podrobnější vyhledávání mezi nimi. Podrobnější vyhledávání umožňuje (oproti jednoduchému vyhledávání v hlavičce) navíc možnost seřazení vyhledaných citací dle několika kriterií.

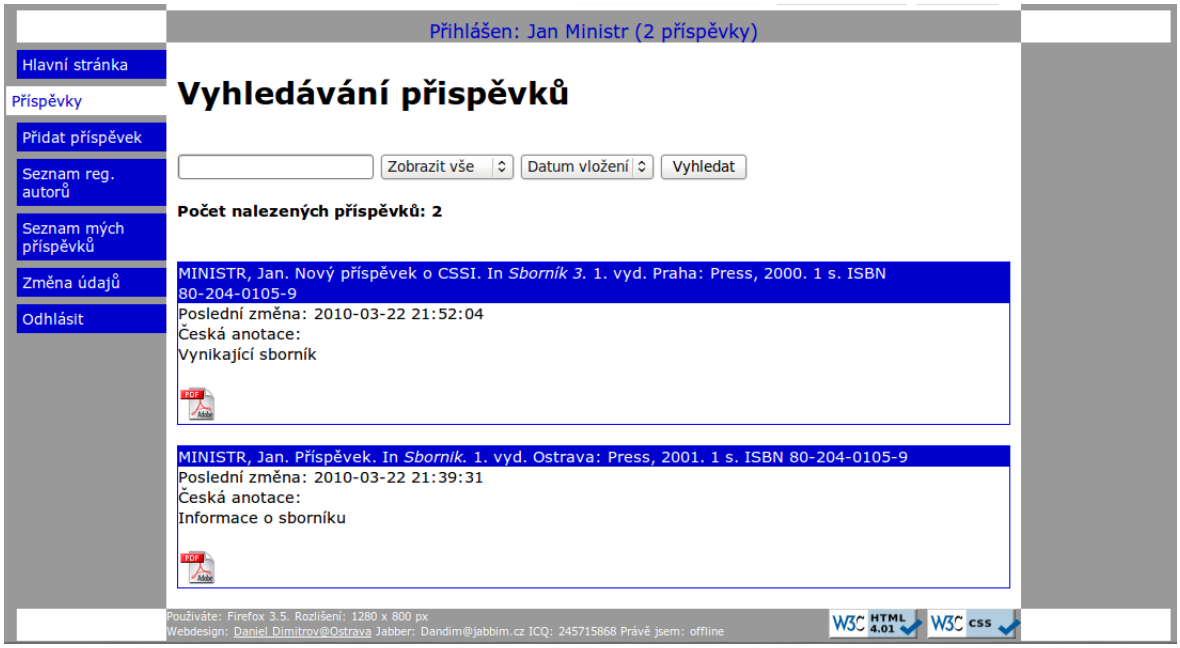
Samotné příspěvky jsou vypsány dle normy ČSN ISO 690.

Typy dokumentů, které lze dle tohoto systému citovat jsou:

- Citace knihy,
- Citace kapitoly knihy,
- Citace časopisu,
- Citace příspěvku ve sborníku,
- Citace webových stránek.

Příspěvky jsou navíc doplněny:

- Poslední změna: Čas poslední změny příspěvku (dle tohoto času je primárně řazeno),
- Česká anotace/Anglická anotace: Při zadávání příspěvku se musí příspěvek doplnit českou či anglickou anotací (případně obojí). Anotace je minimálně 10-ti znaková. Jestli jsou doplněny obojí anotace přednost má anglická,
- Soubor s příspěvkem: Samotný příspěvek lze doplnit souborem, který tento příspěvek obsahuje. Soubor musí být ve formátu PDF a maximální velikost souboru je 20MB.



Obrázek 4.9: Příspěvky

4.3.5 Přidat příspěvek

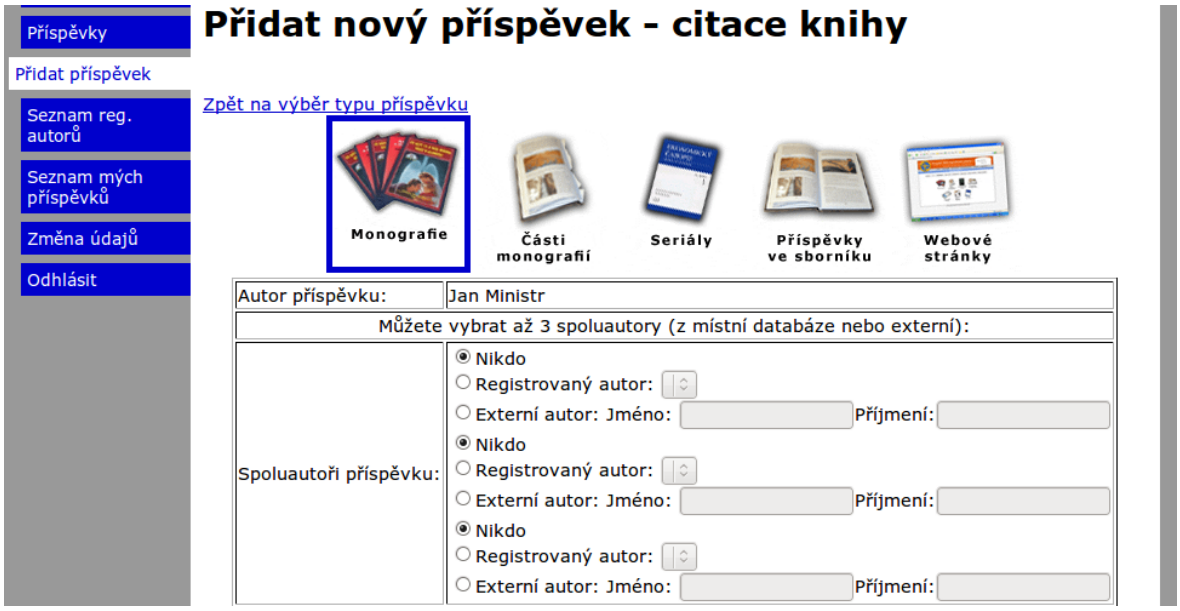
Přidat příspěvek je velmi jednoduchý proces a skládá se z výběru typu dokumentu, vybrání autorů, doplnění informací o příspěvku a nakonec přidání souboru s příspěvkem.

Nejprve je zapotřebí vybrat typ citovaného dokumentu. Možné typy dokumentů jsou zobrazeny v následujícím obrázku 4.10.

Obrázek 4.10: Vybrání typu příspěvku

Samozřejmě u každého typu je souhrn atributů o příspěvku jiný, ale hodně podobný. Po vybrání typu citovaného dokumentu dochází k zapisování informací o příspěvku.

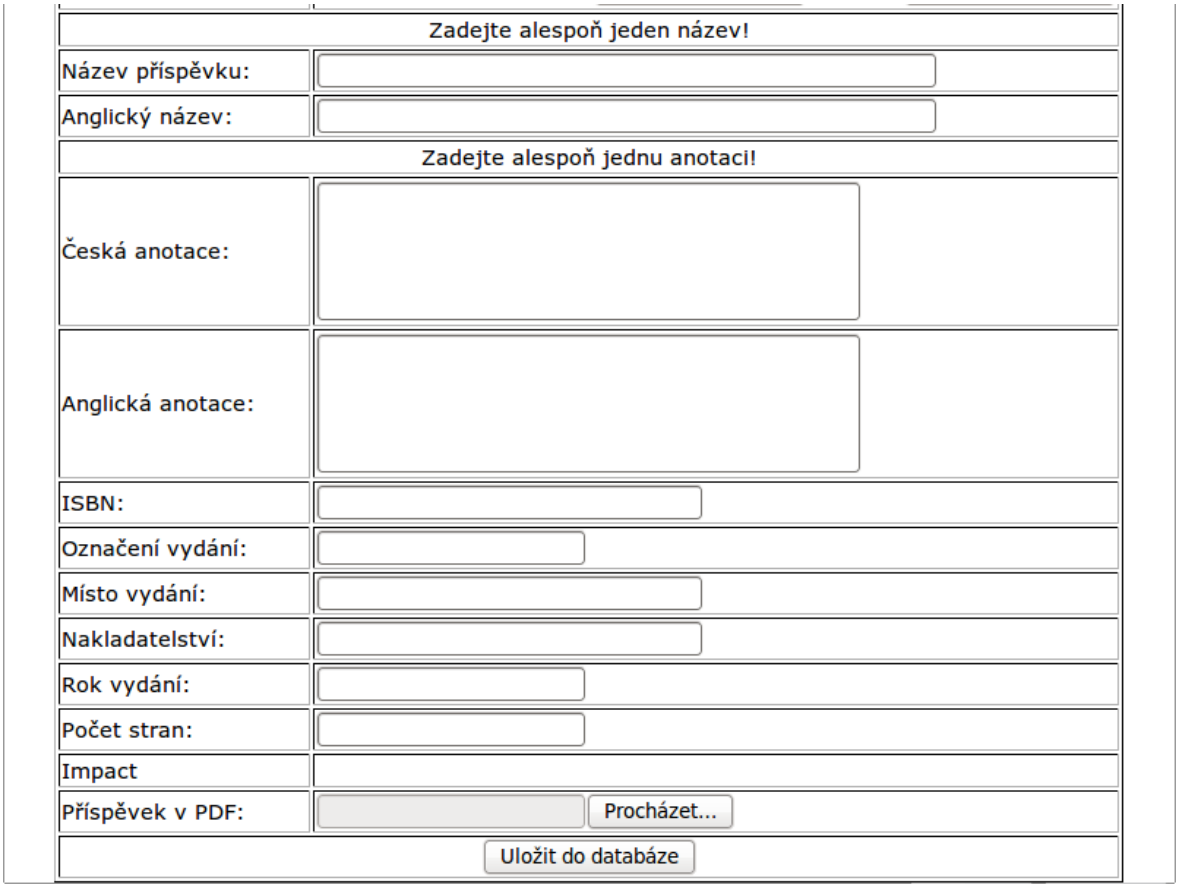
Jako první, u každého příspěvku, musíme vybrat spoluautory příspěvku. Máme možnost jich vybrat až 3, přičemž můžeme volit spoluautora z místní databáze či zadáme nového (bude použit jen v tomto příspěvku).



Obrázek 4.11: Přidání nového příspěvku

Po zadání případných spoluautorů přijde na řadu zadávání informací o příspěvku.

Všechny tyto informace jsou ošetřeny a kontrolovány na případné chyby zadavatele příspěvku. Případné nedostatky jsou uživateli ihned hlášeny pomocí chybového hlášení. Takto je ošetřeno vkládání příspěvků, zamezující chyby uživatele a zvyšující úroveň věrohodnosti a pravdivosti citací.



Obrázek 4.12: Vyplnění atributů příspěvku

Všechna pole jsou povinná, samozřejmě až na nepovinný soubor. Po odeslání správně vyplněného formuláře je okamžitě přidán do databáze a lze jej zobrazit na stránce Příspěvky.

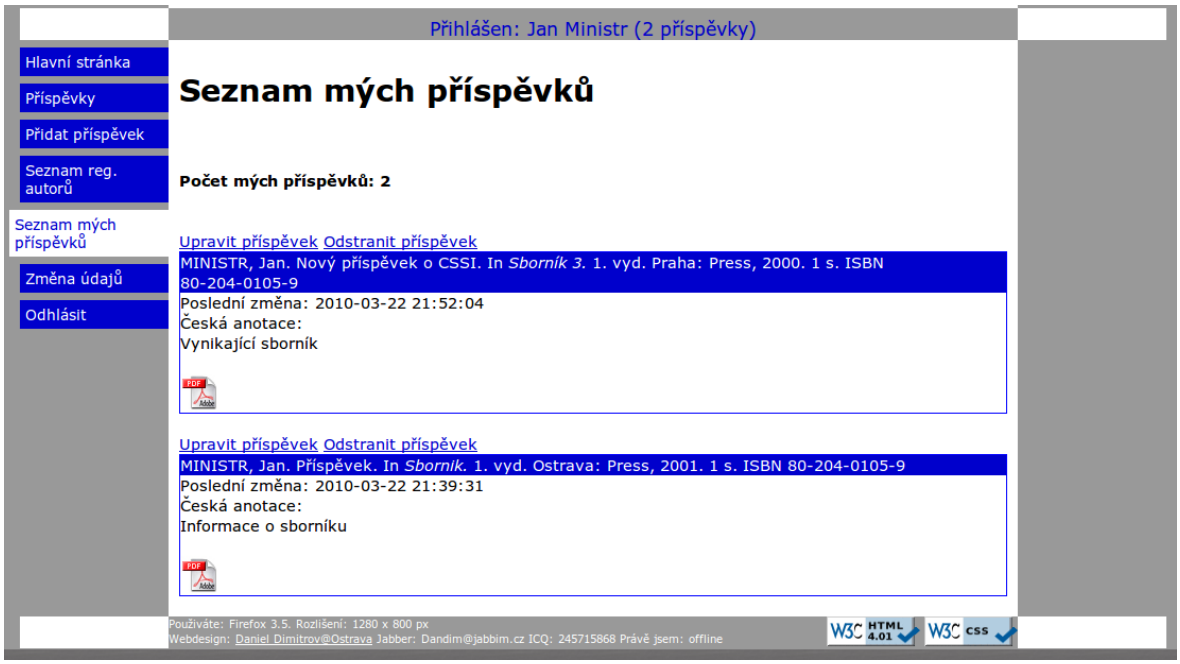
4.3.6 Seznam registrovaných autorů

V této sekci nalezneme seznam autorů příspěvků, který obsahuje kontaktní informace o uživatelích systému. Tento seznam je doplněn i o spoluautory příspěvků použitých v příspěvcích.

Obrázek 4.13: Seznam registrovaných uživatelů

4.3.7 Seznam mých příspěvků

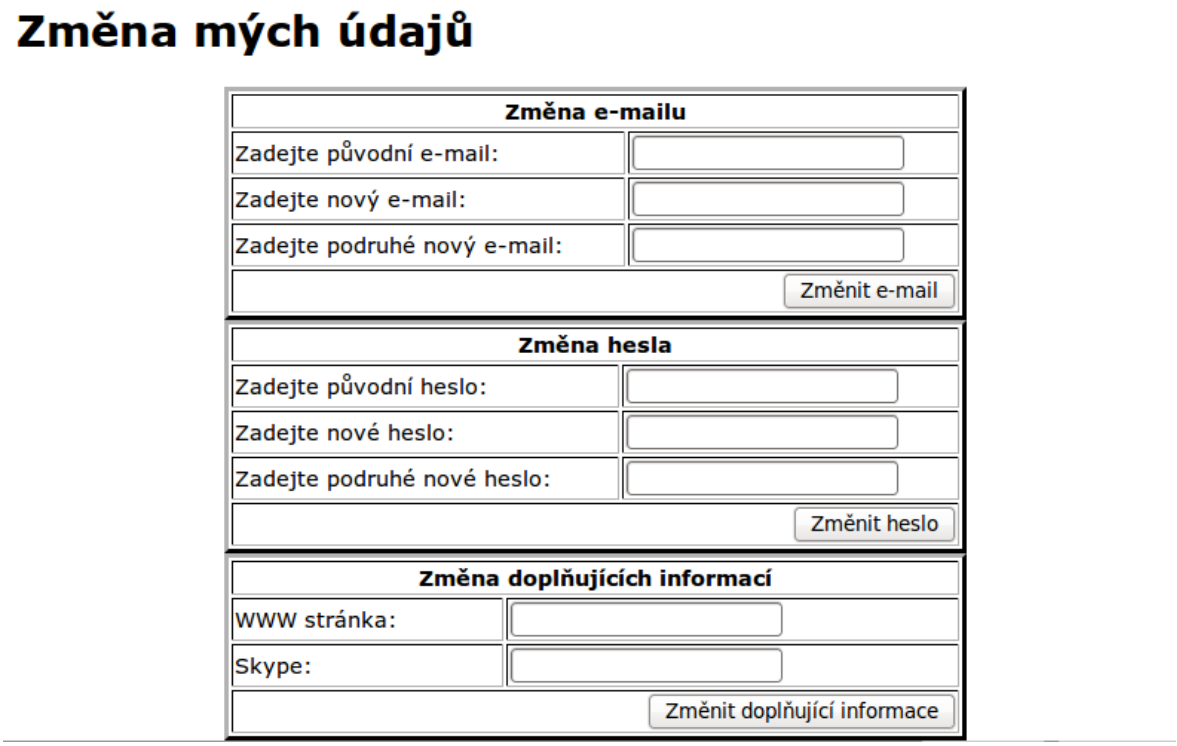
Seznam mých příspěvků obsahuje jen uživatelovy příspěvky, takže na první pohled je struktura stejná jako sekce Příspěvky. V této sekci lze však příspěvky upravovat či mazat.



Obrázek 4.14: Seznam mých příspěvků

4.3.8 Změna údajů

Samozřejmě lze informace zadané při registraci změnit. Změna se provádí kliknutí na položku Změna údajů v menu. Změnit lze e-mail, heslo, www stránku či skype.



Obrázek 4.15: Možnost změny údajů

4.3.9 Odhlásit

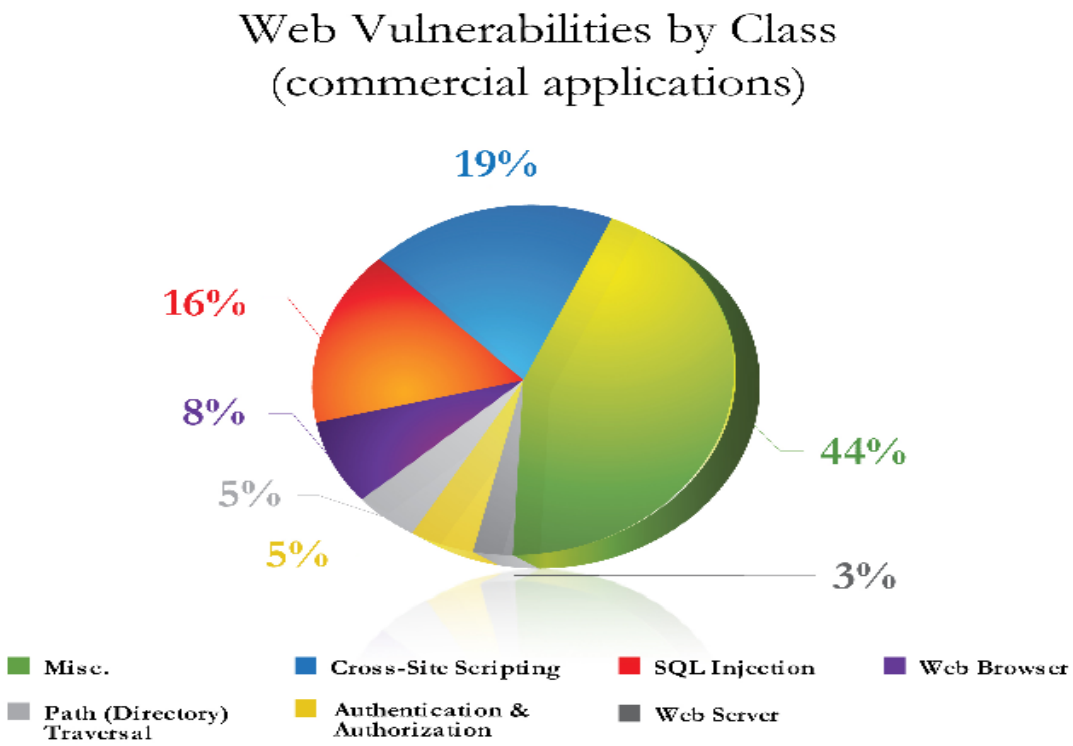
Poslední položkou menu představuje odhlášení ze systému. Korektní odhlášení lze provést jen po kliku na položku menu Odhlásit (pomocí PHP scriptu dojde k smazání obsahu proměnné session) nebo eventuálně zavřením prohlížeče (dojde k vymazání cookies uložených u uživatele v PC). Jinak uživatel zůstane pořád přihlášený a při další návštěvě bude automaticky přihlášen. V systému je nastavena expirace cookies na hodnotu 0. K smazání tedy dojde až po zavření prohlížeče.

Takto probíhá smazání session na straně serveru (kliknutím na Odhlásit):

```
if ($_GET["odkaz"]=="odhlasit"){
    if (isset($_SESSION["id"])) {
        unset($_SESSION["id"]);
        session_destroy(); } header('Location: ./index.php'); }
```

4.4 Bezpečnostní algoritmy

Na závěr popisu realizace systému je třeba představit bezpečnostní hrozby, které by mohly ohrozit systém a anonymitu uživatelů, a bezpečnostní opatření systému. Bezpečnostní hrozby je důležité průběžně sledovat a přizpůsobovat (zdokonalovat) systém. Úkolem této kapitoly je seznámení s nejběžnějšími hrozbami napadajícími webové aplikace a navrhnutí ochrany proti nim. Společnost Cenzic, Inc., zabývající se bezpečností webových aplikací, průběžně vydává zprávy o zranitelnostech aplikací na internetu. Dle těchto zpráv patří mezi nejvíce využívané hrozby: SQL Injection a Cross-site Scripting. [7]

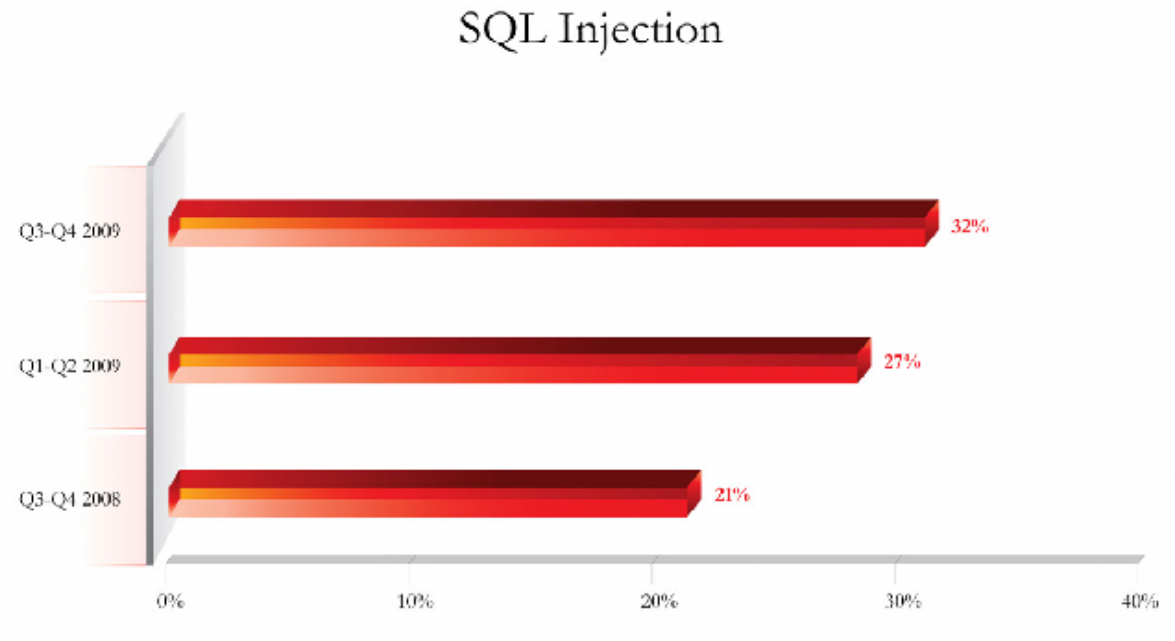


Obrázek 4.16: Rozdělení zranitelností aplikací dle společnosti Cenzic [7]

4.4.1 SQL Injection

Technika prolomení bezpečnosti systému vložením útočníkova SQL dotazu pomocí neošetřených vstupů. Velmi nebezpečná technika umožňující neoprávněný vstup do systému či dokonce operace nad databází (například smazání tabulky, vložení záznamů). Funguje na principu vložení části SQL příkazu do jakékoliv vstupního pole aplikace, která následně provede SQL příkaz doplněný o část SQL příkazu ze vstupního pole. Pokud aplikace nebezpečně upravený SQL dotaz vykoná je aplikace nebezpečná. [2]

Dle společnosti Cenzic, zabývající se webovou bezpečností, je SQL Injection stále největší bezpečnostní hrozbou i když je ochrana vcelku snadná. Na obrázku 4.16 lze vidět rostoucí zranitelnost webových aplikací. [2]



Obrázek 4.17: Rostoucí tendence SQL Injection zranitelnosti dle Cenzic [7]

Příklad SQL Injection útoku [2]:

```
<?php
// SQL dotaz
$query = "SELECT * FROM users WHERE user='{$_POST['username']}' AND
password='{$_POST['password']}'";
mysql_query($query);

// Nezkontrolujeme vstupní hodnoty a uživatel může jako heslo zadat:
$_POST['username'] = 'aidan';
$_POST['password'] = "' OR ''='";

// SQL dotaz se vykoná
echo $query;
?>
//Výsledný SQL dotaz, který se vykoná a umožní vstup díky platné podmínce ' '=' ':
SELECT * FROM users WHERE user='aidan' AND password='' OR ' '=' '
```

Obrana proti útokům [2]:

- nastavení striktních práv uživatele databáze,
- ošetření vstupních hodnot, například funkcí mysql_real_escape_string.

Příklad zabezpečení proti SQL Injection [2]:

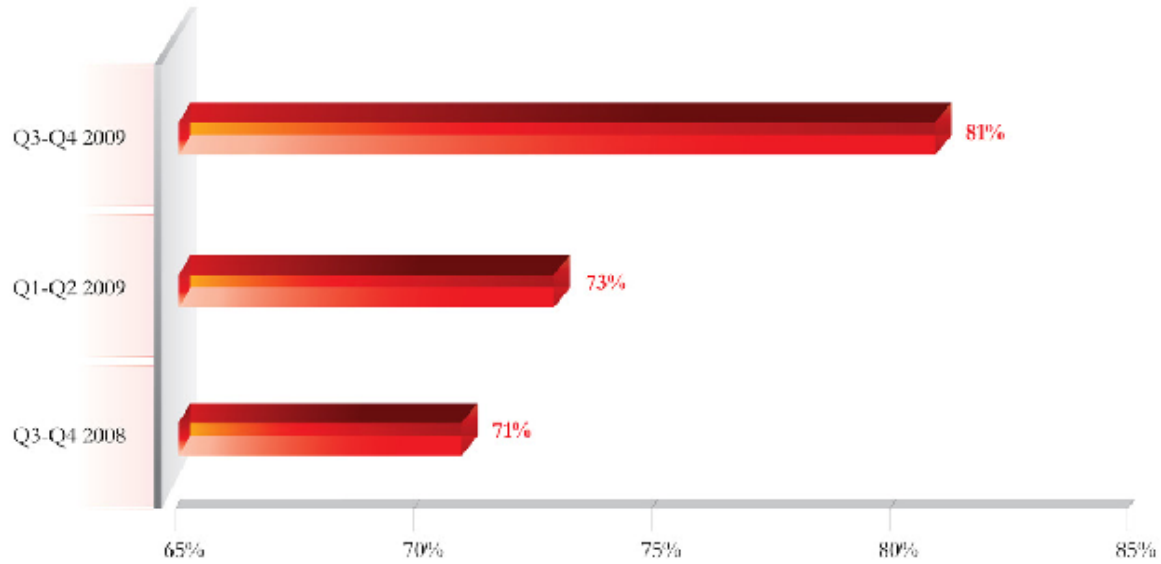
```
foreach ($_POST as $key => $value) {
   $_POST[$key] = mysql_real_escape_string($value); }
```

Použití těchto dvou řádku kódu PHP zabezpečí úpravu prvků (textových řetězců) asociativního pole $_POST na bezpečné řetězce pro použití na vykonání SQL dotazu. Úprava spočívá například v tom, že před apostrof v textu přidá zpětné lomítko. [2]

4.4.2 Cross-site Scripting

Mezi další velmi časté narušení webových aplikací probíhá prostřednictví Cross-site Scripting (XSS) útoku. Tento útok je podobný s předchozím útokem, opět dochází ke vložení zákeřného kódu do vstupního pole či proměnné v URL. Tentokrát jde o JavaScript kód a nebezpečné je právě neošetření vstupního řetězce a následné vložení řetězce z vstupního pole do HTML výstupu aplikace. Tímto dojde k provádění JavaScript kódu na stránce. [2,23]

Dle společnosti Cenzic jde o další rychle rostoucí zranitelnost, dokazující jejich výroční zpráva. Graf z této zprávy je na obrázku 4.17.



Obrázek 4.18: Graf Cross-site Scripting zranitelnosti dle Cenzic [7]

Příklad Cross-site Scripting útoku [2]:

Mějme smyšlenou webovou stránku a útočník vloží toto URL do svého prohlížeče:

```
http://www.domena.cz/index.php?text=Neco<script>alert('Toto je XSS útok.');</script>
```

Smyšlená webová stránka je nedokonalá a součástí kódu je i tento příkaz, vypisující proměnnou *nadpis*:

```
<?php echo $_GET['text']; ?>
```

Tímto by došlo k generování výsledné webové stránce s vloženým JavaScript kódem.

Obrana pro útokům [2]:

- V žádném případě nevypisovat do HTML kódu neošetřený vstup (URL, input, textarea),
- Vstup je zapotřebí ošetřit pomocí funkce htmlspecialchars, která vyhledá v řetězci nebezpečné znaky (“<“,“>“,“&“,“‘“) a převede je na bezpečné entity HTML (“<“,“>“,“&“,“'“).

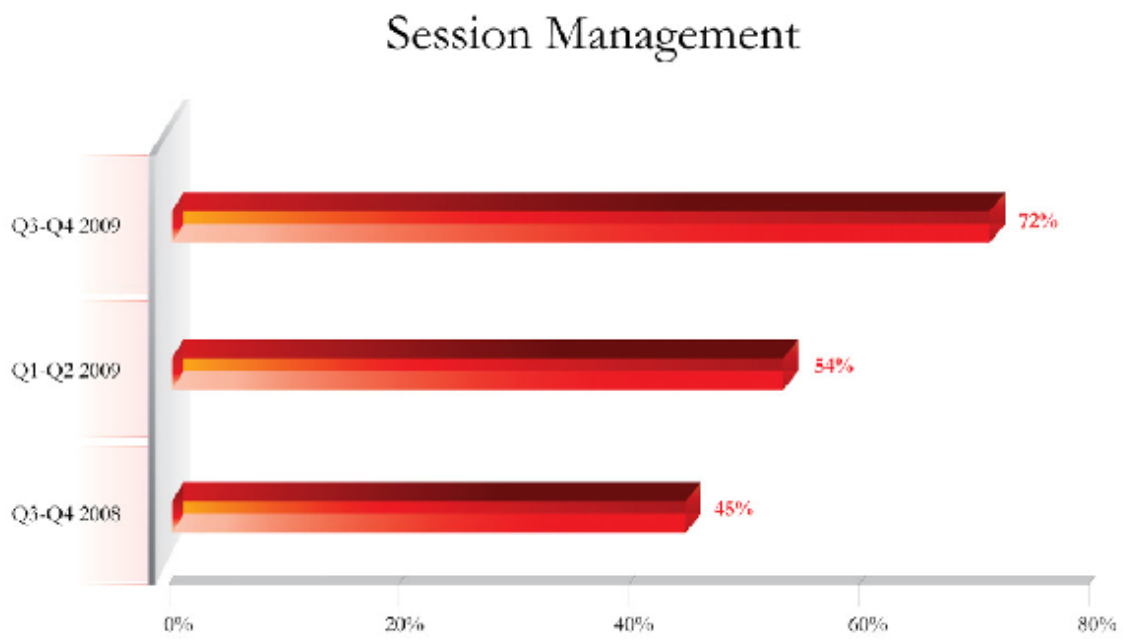
Příklad zabezpečení proti Cross-site Scripting [2]:

```
foreach ($_POST as $key => $value) {
   $_POST[$key] = htmlspecialchars($value,ENT_QUOTES, 'UTF-8'); }
```

4.4.3 Session hijacking

Technika ukradení SID (Session ID, identifikátor session) konkrétního uživatele, vzniklé při vytvoření SESSION uživatele. Útočník využívající Session hijacking se snaží získat platné SID a využít jej. Využitím platného SID získá přístup do aplikace, což je zcela nepřístupné. Při vytvoření SESSION dojde k vytvoření SID, které je uloženo na straně klienta v podobě cookies a je neustále posíláno v HTTP hlavičce. [2,22]

Na straně serveru je zabezpečení SESSION relativně bezpečné (důležité je nastavení serveru a udržovat operační systém a PHP aktuální), ale horší je to straně klienta či na cestě mezi serverem a klientem. Ochrana při přenosu je jediná a to šifrování pomocí SSL. U klienta je situace horší, cookies jsou malé soubory uložené v počítači klienta nešifrovaně. A proto zjištění hodnoty cookies je snadnou cestou získat platné SID. [22]



Obrázek 4.19: Vývoj Session Management zranitelnosti dle Cenzic [7]

Obrana proti útokům [2]:

- Využívat nejnovější PHP, Apache,
- Využívat PHP funkci session_regenerate_id() pro změnu SID,
- Použít šifrovanou komunikaci pomocí SSL,
- V session použít nějaký bezpečnostní ověřovací prvek. Například IP adresu nebo informace o prohlížeči klienta. Tento prvek ověřovat při každém pohybu v systému.

Příklad zabezpečení proti Session hijacking [2]:

```
session_regenerate_id(true);
```

5 Zhodnocení navrhovaného řešení

Navrhovaný informační systém na správu příspěvků je ve finální verzi umístěn na adrese http://www.cssi-morava.cz/new/sprava-citaci/ .

Vytvořený systém splňuje zadání projektu zadavatele a je připraven na nasazení do „ostrého“ provozu. Samozřejmě mezi zadání byla i podpora autora systému a tak bude systém dále rozvíjen, upravován, doplňován a zlepšován na přání uživatelů či zadavatele. Systém zavádí centralizaci správy citací a nabízí tak registrovaným uživatelům minimálně jednu z těchto výhod:

- Jednotný systém správy citací určený jen registrovaným uživatelům. Tímto dochází ke zvýšení důvěryhodnosti příspěvků i samotných uživatelů,
- Jednoduchost zadání nového příspěvku a okamžité vystavení ostatním uživatelům,
- Uživatel může citovat knihu, kapitolu knihy, časopis, sborník či webové stránky,
- Citace uživatelů jsou vypisovány dle aktuální normy ČSN ISO 690,
- Možnost přidání spoluautora z registrovaných uživatelů,
- Možnost přidání anotace i v anglickém jazyku a rozšířit informace o příspěvku,
- Jednoduché a rychlé citování ostatních uživatelů,
- Systém nabízí přehled o registrovaných uživatelích,
- Rychlé či detailnější vyhledávání příspěvků dle kriterií,
- Veškeré data jsou průběžně zálohována a chráněna před zneužitím.

Systém je navíc nadále administrován a tím zajištěn dohled nad fungováním systému. Celý systém byl ručně napsán v jazyku PHP, což umožňuje flexibilní přidání funkcí či změnu stávajících.

6 Závěr

Projekt měl jasný cíl, který jsem se snažil nejen splnit, ale i předčit. Práce obsahuje 8 kapitol a několik podkapitol až do třetí úrovně.

V kapitole 1 je úvod, kde jsem identifikoval předpokládané vlastnosti budoucího systému a stanovil cíl projektu.

Kapitola 2 obsahuje výklad teoretických pojmů a technologií použitých při samotné práci na systému. Kapitola se postupně zabývá teoretickými pojmy systémové integrace a následně popisuje architekturu LAMP serveru. Nakonec jsou stručně popsány internetové technologie, které byly použity při vývoji systému.

Kapitola 3 se zabývá analýzou současného stavu problémové oblasti, kdy jsem vytvářel zcela nový systém, který uživatelům scházel.

Kapitola 4 se zabývá popisem práce se systémem s důrazem na jeho bezpečnost. V podkapitole o bezpečnostních algoritmech jsem se snažil nastínit můj zájem o bezpečnost aplikace a představení bezpečnostních hrozeb.

V kapitole 5 jsou představeny hlavní přínosy vytvořeného informačního systému pro samotného uživatele systému.

Samotný informační systém splňuje cíl, předpokládané vlastnosti určené zadavatelem a nakonec další neodmyslitelnou vlastností systému je možnost dále tento sytém rozšiřovat o další funkce. Systém lze přizpůsobovat dle aktuálních potřeb uživatelů, což je vlastnost systému velmi důležitá. Mezi uvažovanými náměty o rozšíření jsou: zlepšení designu stránek, vytvoření backend pro administrátora, rozšířit systém o soukromé zprávy či veřejné fórum. Každé další rozšiřování musí schválit zadavatel.

Cílem tohoto projektu bylo vytvoření informačního systému s ohledem na bezpečnost a spolehlivost. Tento cíl se mi podařil splnit. Systém jsem vytvořil s požadovanými funkcemi a upravoval pro svou stabilitu a bezpečnost, které byla věnovaná velká váha při samotné práci na projektu.

Mojí další snahou bude systém pomalu rozšiřovat o další užitečné a žádané funkce tak, aby nebyla porušena stabilita a bezchybná funkčnost systému.

7 Seznam použité literatury

7.1 Knihy

1. HUB, Miloslav. *Technologie internetu – PHP 5.* 1. vyd. Pardubice: Univerzita Pardubice, 2009. 88 s. ISBN 978-80-7395-163-4.
2. HUSEBY, Sverre H. *Zranitelný kód.* 1. vyd. Brno: Computer Press, a.s., 2006. 207 s. ISBN 80-251-1180-6.
3. KOSEK, Jiří. *PHP : Tvorba interaktivních internetových aplikací*. Praha : Grada Publishing, a.s., 1999. 492 s. ISBN 80-7169-373-1.
4. NARAMORE, Elizabeth, GERNER, Jason, LE SCOUARNEC, Yann, STOLZ, Jeremy, GLASS, Michael K. *PHP5, MySQL, Apache. Vytváříme webové aplikace.* 1. vyd. Brno: Computer Press, a.s., 2006. 813 s. ISBN 80-251-1073-7.
5. SHAH, Steve, SOYINKA, Wale. *Administrace systému Linux.* 4. vyd. Praha: Grada Publishing, a. s., 2007. 428 s. ISBN 978-80-247-1694-7.
6. TVRDÍKOVÁ, Milena. *Aplikace moderních informačních technologií v řízení firmy.* 1. vyd. Praha: Grada Publishing, a.s., 2008. 173 s. ISBN 978-80-247-2728-8.

7.2 Internetové zdroje

7. CENZIC, INC. *Web Application Security Trends Report Q3-Q4, 2009.* [online]. 2010 [cit. 2010-03-28]. Dostupný z WWW: <http://www.cenzic.com/downloads/Cenzic_AppSecTrends_Q3-Q4-2009.pdf>.
8. *Data types.* [online]. 2008 [cit. 2010-01-20]. Dostupný z WWW: <http://dev.mysql.com/doc/refman/5.1/en/data-types.html>.
9. *How the ASF works.* [online]. [cit. 2010-01-15]. Dostupný z WWW: <http://www.apache.org/foundation/how-it-works.html>.
10. JANOVSKÝ, Dušan. *CSS styly - úvod.* [online]. [cit. 2010-01-30]. Dostupný z WWW: <http://www.jakpsatweb.cz/css/css-uvod.html>.

11. JANOVSKÝ, Dušan. *Úvod do JavaScriptu.* [online]. [cit. 2010-01-30]. Dostupný z WWW: <http://www.jakpsatweb.cz/javascript/javascript-uvod.html>.

12. JANOVSKÝ, Dušan. *Základy HTML.* [online]. [cit. 2010-01-27]. Dostupný z WWW: <http://www.jakpsatweb.cz/zaklady-html.html>.

13. *January 2010 Web Server Survey.* [online]. 2010 [cit. 2010-01-11]. Dostupný z WWW:<http://news.netcraft.com/archives/2010/01/07/january_2010_web_server_survey.html>

14. *mysql — The MySQL Command-Line Tool.* [online]. 2008 [cit. 2010-01-20]. Dostupný z WWW:<http://dev.mysql.com/doc/refman/5.1/en/mysql.html>.

15. *MySQL Workbench Introduction.* [online]. 2008 [cit. 2010-01-20]. Dostupný z WWW:<http://dev.mysql.com/doc/workbench/en/wb-intro.html>.

16. *Overview of the GNU System.* [online]. 1996-2010 [cit. 2010-01-15]. Dostupný z WWW: <http://www.gnu.org/gnu/gnu-history.html>.

17. *PostgreSQL-About.* [online]. 1996-2010 [cit. 2010-01-15]. Dostupný z WWW: <http://www.postgresql.org/about/>.

18. *PhpMyAdmin.* [online]. 2003-2010 [cit. 2010-01-15]. Dostupný z WWW: <http://www.phpmyadmin.net/home_page/index.php>.

19. RETHANS, Derick. *What LAMP can do for you.* [online]. 15.4.2005 [cit. 10.2.2010]. Dostupný z WWW:<http://talks.php.net/show/lamp-ikt-grenland>.

20. *The InnoDB Storage Engine.* [online]. 2008 [cit. 2010-01-20]. Dostupný z WWW: <http://dev.mysql.com/doc/refman/5.1/en/innodb.html>.

21. *The MyISAM Storage Engine.* [online]. 2008 [cit. 2010-01-20]. Dostupný z WWW: <http://dev.mysql.com/doc/refman/5.1/en/myisam-storage-engine.html>.

22. TICHÝ, Jiří. *Session hijacking aneb ukradení session ID.* [online]. 12.9.2008 [cit. 28.3.2010]. Dostupný z WWW:<http://www.phpguru.cz/clanky/session-hijacking>.

23. VRÁNA, Jakub. *Cross Site Scripting.* [online]. 23.11.2005 [cit. 2010-03-28]. Dostupný z WWW:<http://php.vrana.cz/cross-site-scripting.php>.

24. *What can PHP do?* [online]. 2001-2009 [cit. 2010-01-20]. Dostupný z WWW: <http://cz.php.net/manual/en/intro-whatcando.php>.

25. *What is MySQL?* [online]. 2008 [cit. 2010-01-11]. Dostupný z WWW: <http://dev.mysql.com/doc/refman/5.1/en/what-is-mysql.html>.

8 Seznam zkratek

LAMP - Linux+Apache+MySQL+PHP

LAPP - Linix+Apache+PostgreSQL+PHP

SHA - Secure Hash Algorithm

CSS - Cascading Style Sheets

SSL - Secure Socket Layer

SSH - Secure Shell

DNS - Domain Name System

FTP - File Transfer Protocol

GPL - General Public License

HTML - Hypertext Markup Language

PHP - Hypertext Preprocessor

SEO - Search Engine Optimization

SQL - Structured Query Language

TCP/IP - Transmission Control Protocol / Internet Protocol

URL - Uniform Ressource Locator

W3C - World Wide Web Consortium

GUI - Graphical User Interface

Prohlášení o využití výsledků bakalářské práce

Prohlašuji, že

- jsem byl seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo;
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně, ke své vnitřní potřebě, bakalářskou práci užít (§ 35 odst. 3);
- souhlasím s tím, že bakalářská práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO a jeden výtisk bude uložen u vedoucího bakalářské práce. Souhlasím s tím, že bibliografické údaje o bakalářské práci budou zveřejněny v informačním systému VŠB-TUO;
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;
- bylo sjednáno, že užít své dílo, bakalářskou práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Ostravě dne 30.4.2010

……

Daniel Dimitrov

Adresa trvalého pobytu studenta:

Okrajní 16

Ostrava 2

71000

9 Přílohy

Součástí práce je přiložené DVD.